ETH zürich

Demystifying Resolver Work: A Formal Perspective on DNS Resolver Performance Vulnerabilities

Liwen Xu, Doctoral Researcher, ETH Zürich (liwen.xu@inf.ethz.ch)

Joint work with: Huayi Duan (HKUST-GZ), Zechao Cai, and Adrian Perrig (ETH Zürich)

armasuisse

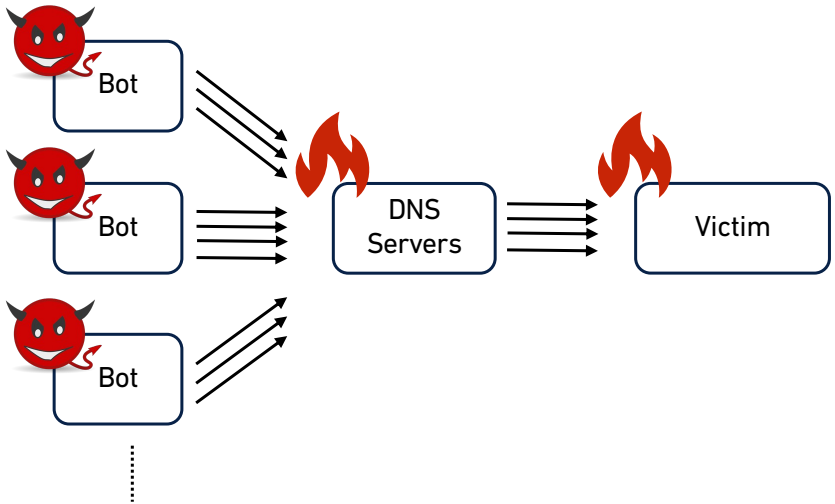
ZISC Zurich Information Security & Privacy Center

Swiss National Science Foundation

HASLERSTIFTUNG

Classes of DoS Attacks Targeting Resolvers

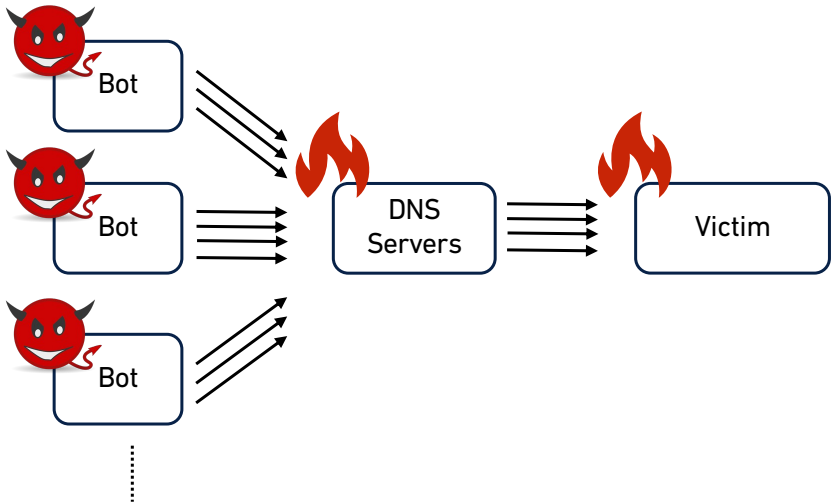
Traditional Volumetric DDoS



Many queries
Large traffic volume

Classes of DoS Attacks Targeting Resolvers

Traditional Volumetric DDoS



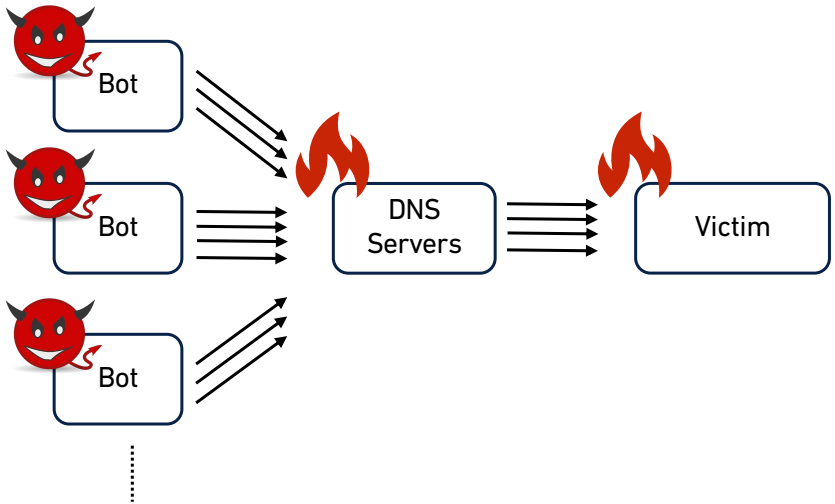
Many queries
Large traffic volume

Resolver Work Amplification

A few queries
Disproportionate work

Classes of DoS Attacks Targeting Resolvers

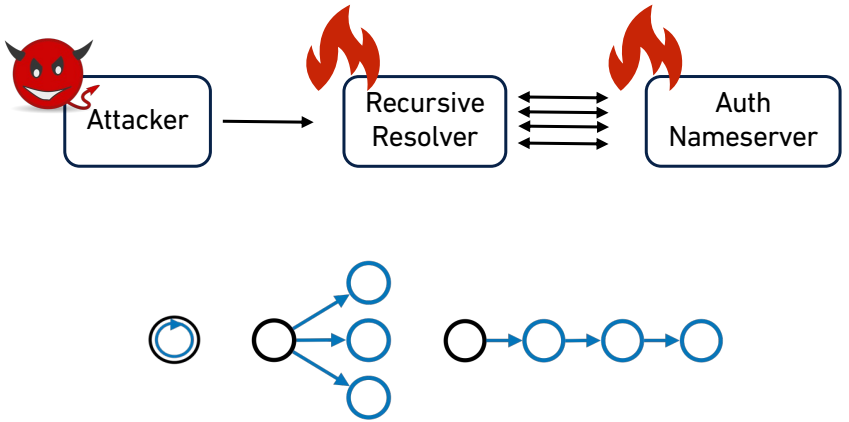
Traditional Volumetric DDoS



Many queries
Large traffic volume

Resolver Work Amplification

(aka, Self-Amplification @ DNS-OARC'43)



A few queries
Disproportionate work

The Problem That Never Got Answered Explicitly To The Full

> The recommended priorities for the resolver designer are:
>
> 1. **Bound the amount of work** (packets sent, parallel processes started) so that a request can't get into an infinite loop or start off a chain reaction of requests or queries with other implementations EVEN IF SOMEONE HAS INCORRECTLY CONFIGURED SOME DATA.
>
> — RFC1034

The Problem That Never Got Answered Explicitly To The Full

> The recommended priorities for the resolver designer are:
>
> 1. **Bound the amount of work** (packets sent, parallel processes started) so that a request can't get into an infinite loop or start off a chain reaction of requests or queries with other implementations EVEN IF SOMEONE HAS INCORRECTLY CONFIGURED SOME DATA.
>
> — RFC1034

What RFC 1034 leaves undefined

- Precise definition of "work”
- Way to measure it
- Guidance on the bound

The Problem That Never Got Answered Explicitly To The Full

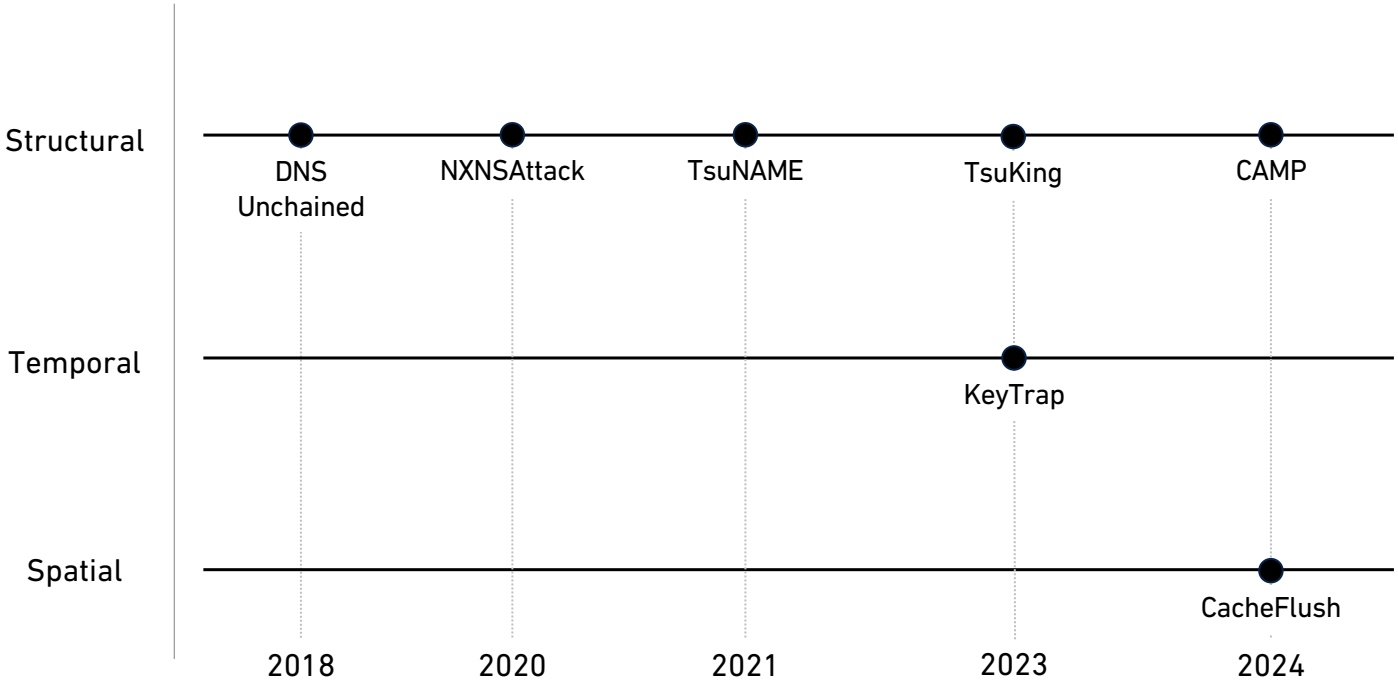
> The recommended priorities for the resolver designer are:
>
> 1. **Bound the amount of work** (packets sent, parallel processes started) so that a request can't get into an infinite loop or start off a chain reaction of requests or queries with other implementations EVEN IF SOMEONE HAS INCORRECTLY CONFIGURED SOME DATA.
>
> – RFC1034

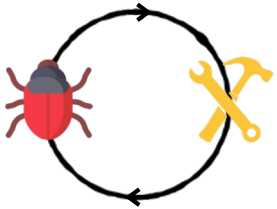
What RFC 1034 leaves undefined

- Precise definition of "work"
- Way to measure it
- Guidance on the bound

What is the maximum amount of work a resolver can perform for a single query?

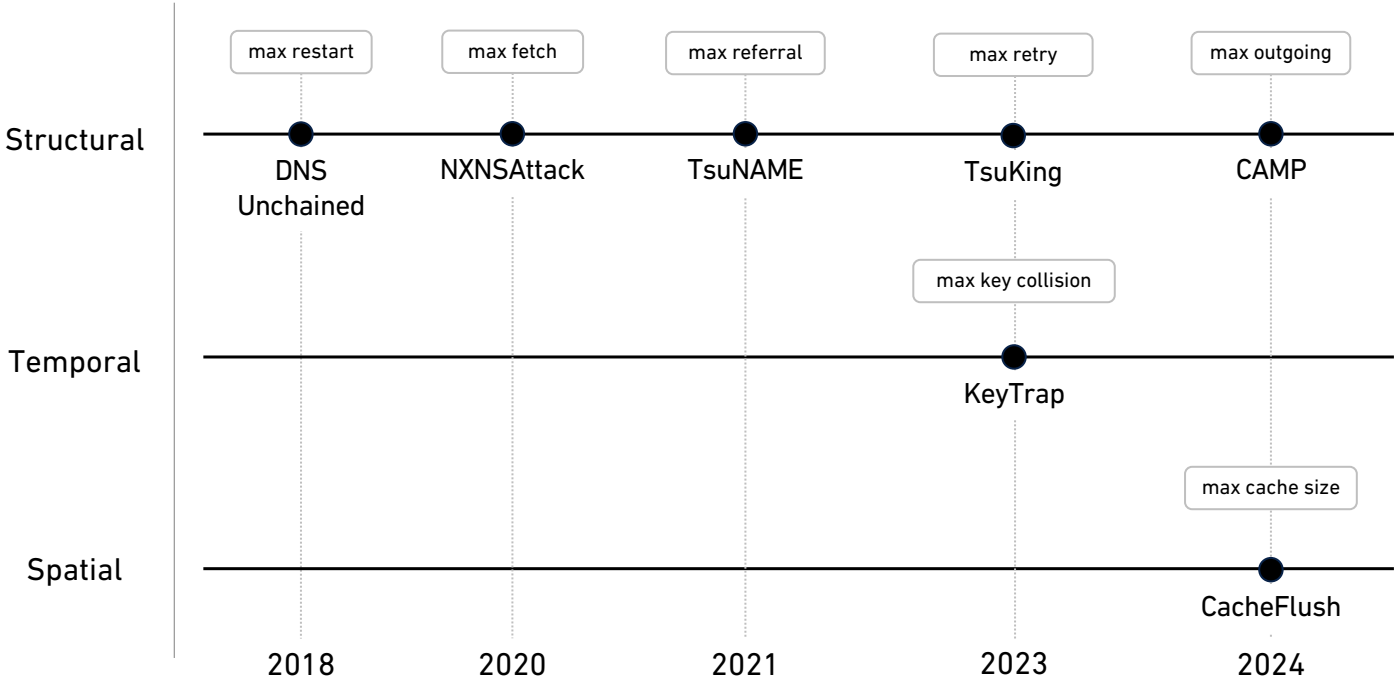
We're Caught In A Break-and-Fix Cycle

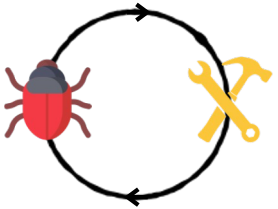




* Taxonomy based on the primary source of introduced complexity

We're Caught In A Break-and-Fix Cycle





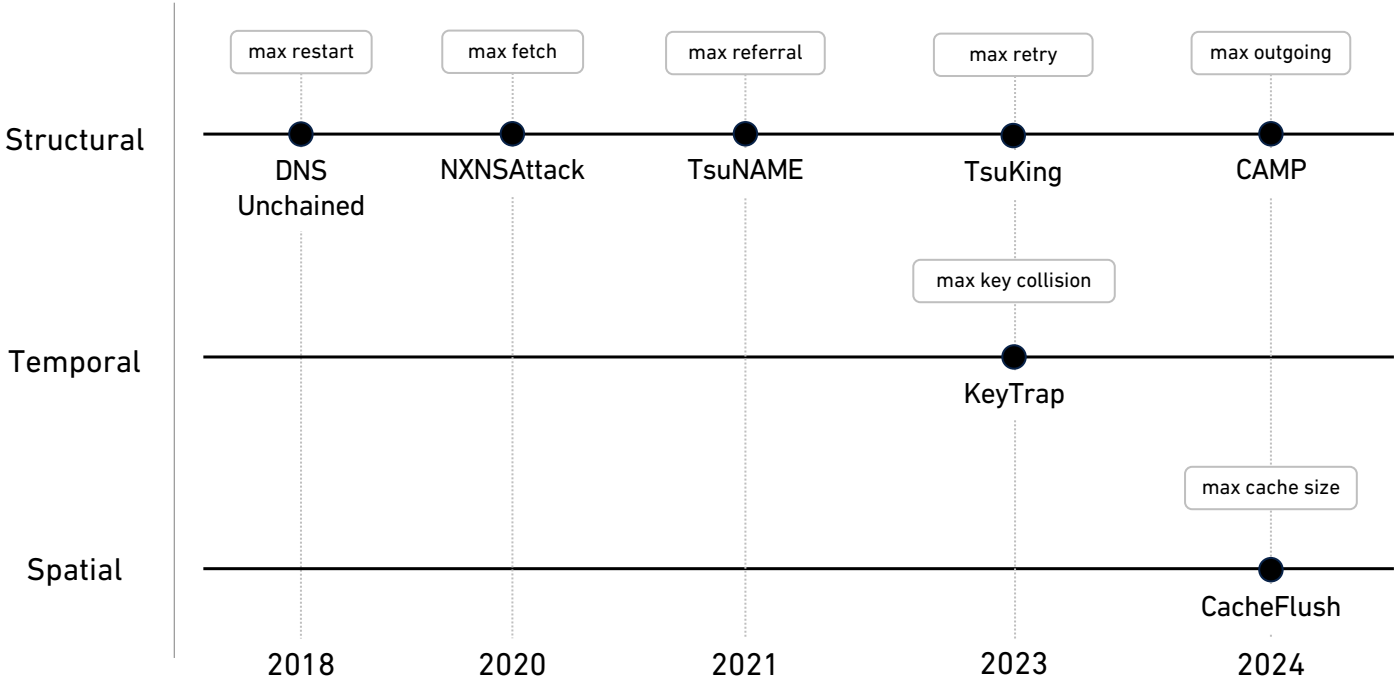
New attack vector

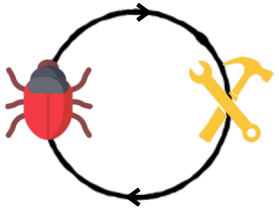
↓

New patch(es)

* Taxonomy based on the primary source of introduced complexity

We're Caught In A Break-and-Fix Cycle





New attack vector

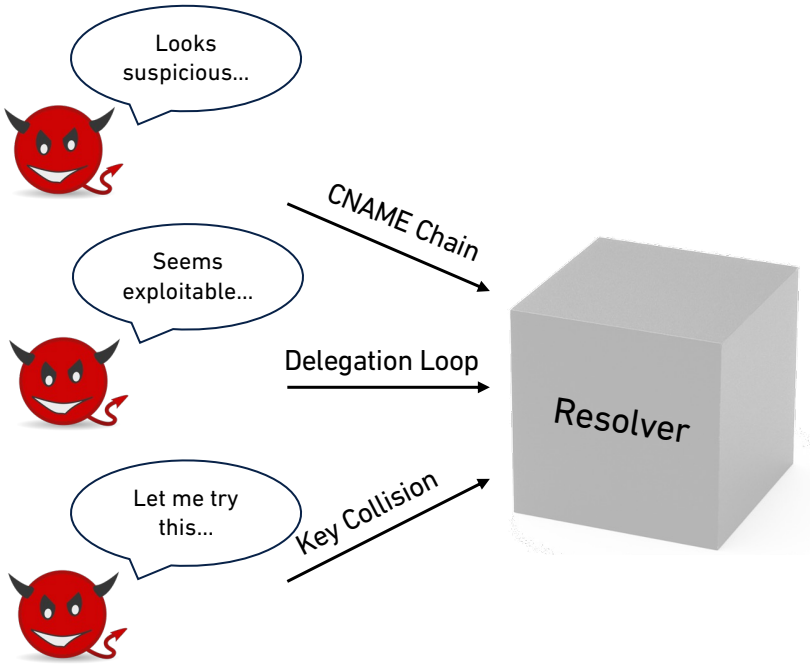
↓

New patch(es)

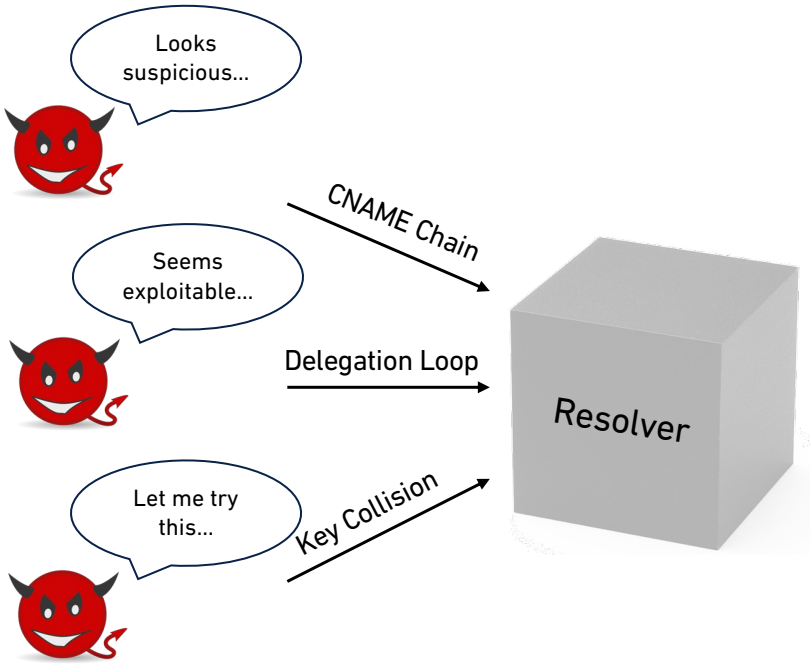
* Taxonomy based on the primary source of introduced complexity

Are there more? How many more?

From Intuition to Formalization

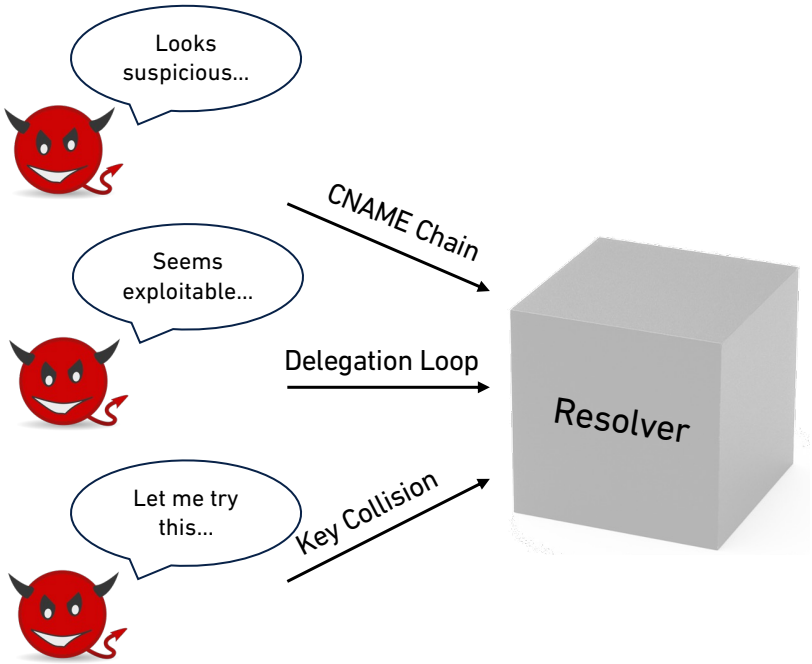


From Intuition to Formalization



Let's go back to where it all started.

From Intuition to Formalization



Let's go back to where it all started.

The top level algorithm has four steps:

1. See if the answer is in local information, and if so return it to the client.

2. Find the best servers to ask.

3. Send them queries until one returns a response.

4. Analyze the response, either:

   a. if the response answers the question or contains a name error, cache the data as well as returning it back to the client.

   b. if the response contains a better delegation to other servers, cache the delegation information, and go to step 2.

   c. if the response shows a CNAME and that is not the answer itself, cache the CNAME, change the SNAME to the canonical name in the CNAME RR and go to step 1.

   d. if the response shows a servers failure or other bizarre contents, delete the server from the SLIST and go back to step 3.

– RFC1034, §5.3.3

Resolution as a Transition System

Algorithm 1: Recursive Resolution Algorithm

Input: A query $Q = (SNAME, STYPE)$ from a client
Output: Resolved answer A or an error response $\perp$

```
   /* Step 1: Check local cache                          */
1  if ∃A = (SNAME, STYPE, RRs) ∈ CACHE then
2      return A to client ;
3      terminate ;
   /* Step 2: Find the best authoritative
      servers                                            */
4  SLIST ← FindBestServers(Q) ;
   /* Step 3: Send queries and wait for a
      response                                           */
5  repeat
6      foreach s ∈ SLIST do
7          Send query Q to server s ;
8      Collect all responses in R ;
9  until ∃R ∈ R such that R ≠ ∅;
   /* Step 4: Case A                                     */
10 if A = (SNAME, STYPE, RRs) ∈ R or R =⊥ (Name
     Error) then
11     CACHE ← CACHE ∪ {A} ;
12     return R to client ;
13     terminate ;
   /* Step 4: Case B                                     */
14 else if NS: D = (SNAME, S) ∈ R and D is valid then
15     CACHE ← CACHE ∪ {D} ;
16     SLIST ← SLIST ∪ {S} ;
17     goto step 2 ;
   /* Step 4: Case C                                     */
18 else if CNAME: C = (SNAME → SNAME′) ∈ R and
     STYPE ≠ CNAME then
19     CACHE ← CACHE ∪ {C} ;
20     Q ← (SNAME′, STYPE) ;
21     goto step 1 ;
   /* Step 4: Case D                                     */
22 else if R indicates a server failure or an invalid response then
23     SLIST ← SLIST \ {s} ;
24     goto step 3
25     or
26     return SERVFAIL to client;
27     terminate ;
```

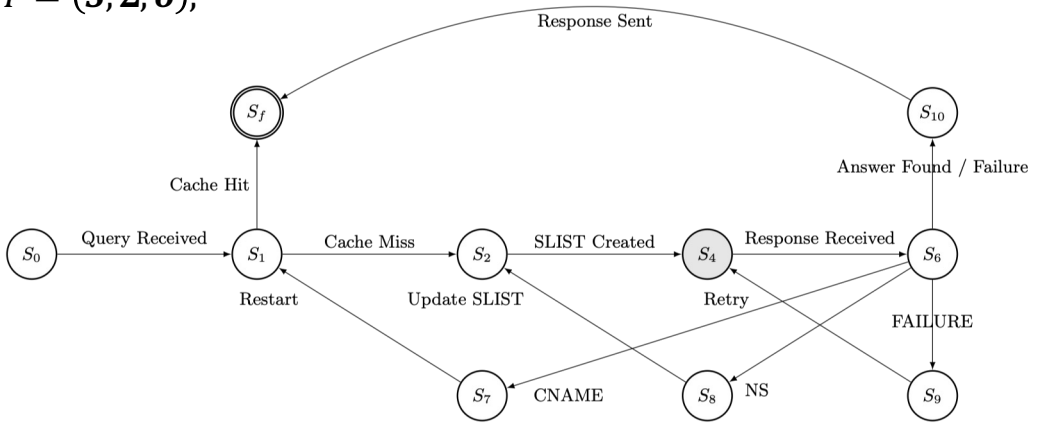
Resolution as a Transition System

Algorithm 1: Recursive Resolution Algorithm

Input: A query $Q = (SNAME, STYPE)$ from a client
Output: Resolved answer A or an error response $\perp$

```
/* Step 1: Check local cache                                   */
if ∃A = (SNAME, STYPE, RRs) ∈ CACHE then
    return A to client ;
    terminate ;
/* Step 2: Find the best authoritative servers                 */
SLIST ← FindBestServers(Q) ;
/* Step 3: Send queries and wait for a response                */
repeat
    foreach s ∈ SLIST do
        Send query Q to server s ;
    Collect all responses in R ;
until ∃R ∈ R such that R ≠ ∅;
/* Step 4: Case A                                              */
if A = (SNAME, STYPE, RRs) ∈ R or R =⊥ (Name Error) then
    CACHE ← CACHE ∪ {A} ;
    return R to client ;
    terminate ;
/* Step 4: Case B                                              */
else if NS: D = (SNAME, S) ∈ R and D is valid then
    CACHE ← CACHE ∪ {D} ;
    SLIST ← SLIST ∪ {S} ;
    goto step 2 ;
/* Step 4: Case C                                              */
else if CNAME: C = (SNAME → SNAME′) ∈ R and STYPE ≠ CNAME then
    CACHE ← CACHE ∪ {C} ;
    Q ← (SNAME′, STYPE) ;
    goto step 1 ;
/* Step 4: Case D                                              */
else if R indicates a server failure or an invalid response then
    SLIST ← SLIST \ {s} ;
    goto step 3
    or
    return SERVFAIL to client;
    terminate ;
```

$$T = (\boldsymbol{S}, \boldsymbol{\Sigma}, \boldsymbol{\delta}),$$

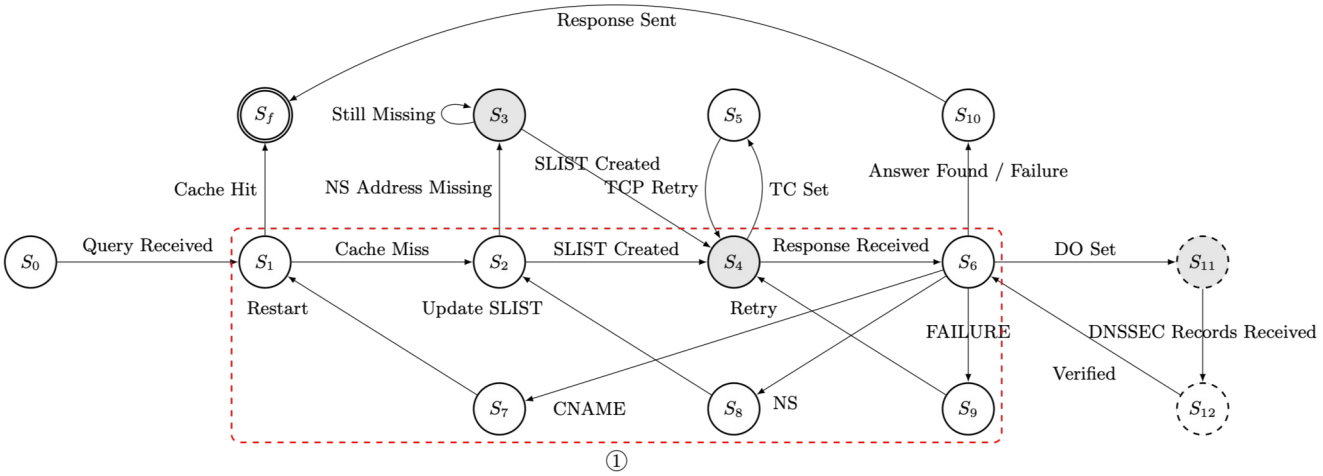


① Resolver Top Level Algorithm

Answer	$\delta(S_6, \sigma_{\text{ANSWER}}) \rightarrow (S_{10})$	Referral	$\delta(S_8, \sigma_{NS}) \rightarrow (S_2)$
Rewrite	$\delta(S_7, \sigma_{CNAME}) \rightarrow (S_1)$	Failover	$\delta(S_9, \sigma_{FAIL}) \rightarrow (S_4)$

State Transition

$T = (\boldsymbol{S}, \boldsymbol{\Sigma}, \boldsymbol{\delta})$,



① Resolver Top Level Algorithm

Answer
$\delta(S_6, \sigma_{\text{ANSWER}}) \rightarrow (S_{10})$

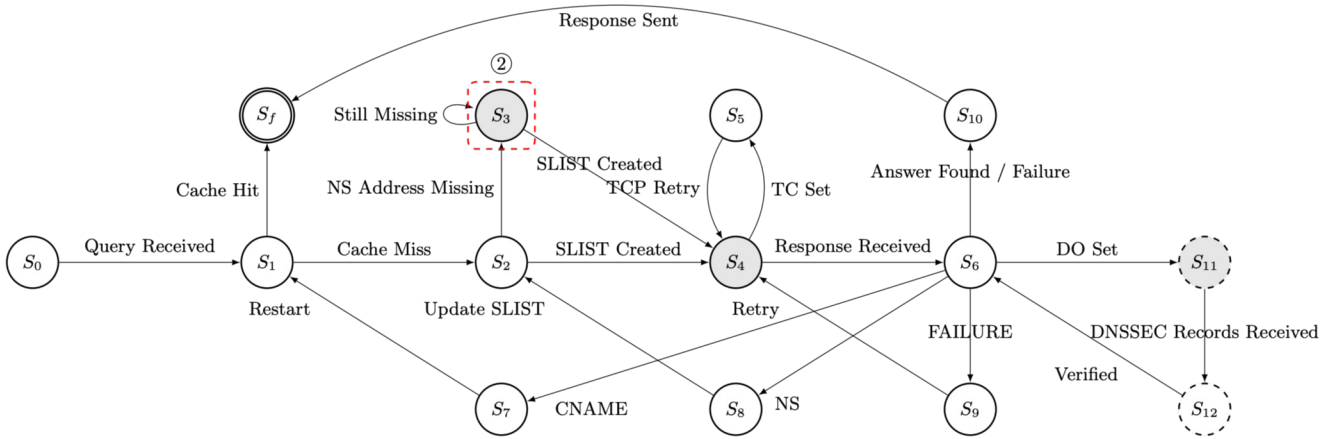
Rewrite
$\delta(S_7, \sigma_{CNAME}) \rightarrow (S_1)$

Referral
$\delta(S_8, \sigma_{NS}) \rightarrow (S_2)$

Failover
$\delta(S_9, \sigma_{FAIL}) \rightarrow (S_4)$

State Transition

$T = (\boldsymbol{S}, \boldsymbol{\Sigma}, \boldsymbol{\delta})$,



① Resolver Top Level Algorithm

Answer
$\delta(S_6, \sigma_{\text{ANSWER}}) \rightarrow (S_{10})$

Rewrite
$\delta(S_7, \sigma_{CNAME}) \rightarrow (S_1)$

Referral
$\delta(S_8, \sigma_{NS}) \rightarrow (S_2)$

Failover
$\delta(S_9, \sigma_{FAIL}) \rightarrow (S_4)$

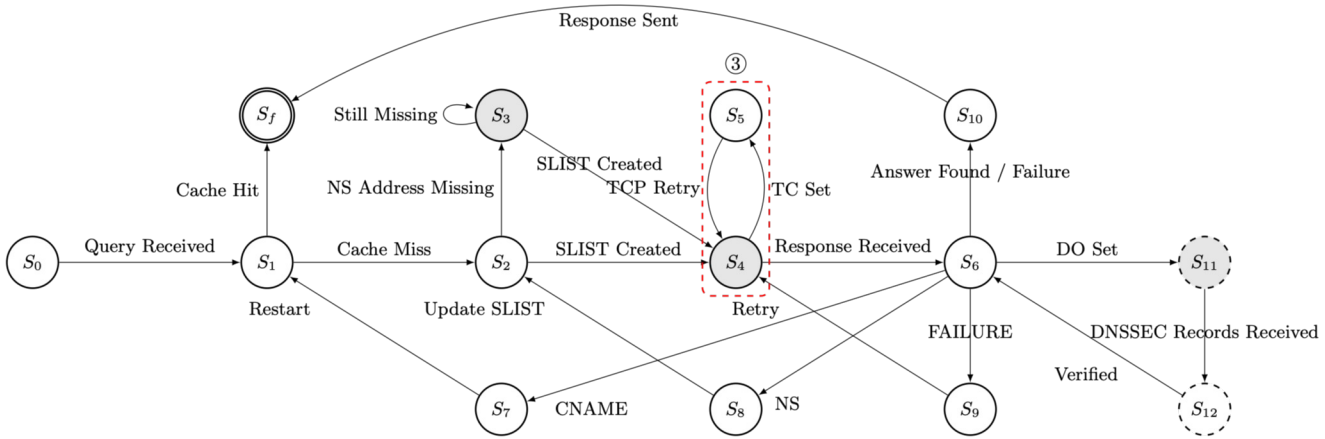
② Missing Nameserver Addresses

$\delta(S_2, \sigma_{MISS}) \rightarrow (S_3)$

$\delta(S_3, \sigma_{MISS}) \rightarrow (S_3)$

State Transition

$T = (\boldsymbol{S}, \boldsymbol{\Sigma}, \boldsymbol{\delta})$,



① Resolver Top Level Algorithm

Answer
$\delta(S_6, \sigma_{\text{ANSWER}}) \rightarrow (S_{10})$

Rewrite
$\delta(S_7, \sigma_{CNAME}) \rightarrow (S_1)$

Referral
$\delta(S_8, \sigma_{NS}) \rightarrow (S_2)$

Failover
$\delta(S_9, \sigma_{FAIL}) \rightarrow (S_4)$

② Missing Nameserver Addresses

$\delta(S_2, \sigma_{MISS}) \rightarrow (S_3)$

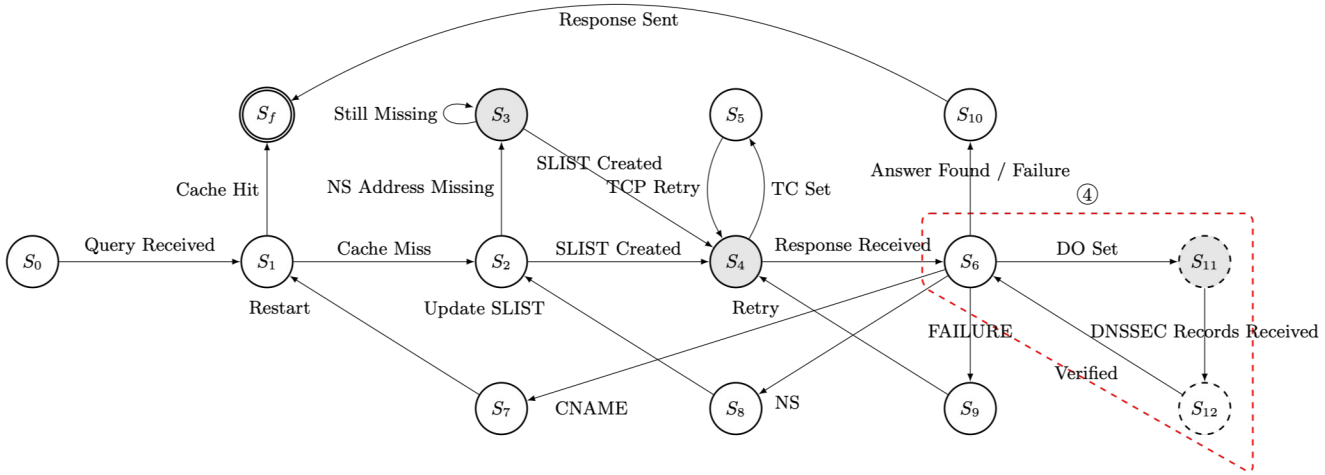
$\delta(S_3, \sigma_{MISS}) \rightarrow (S_3)$

③ TCP Retransmission

$\delta(S_4, \sigma_{TC}) \rightarrow (S_5)$

State Transition

$T = (\boldsymbol{S}, \boldsymbol{\Sigma}, \boldsymbol{\delta})$,



① Resolver Top Level Algorithm

Answer
$\delta(S_6, \sigma_{\text{ANSWER}}) \rightarrow (S_{10})$

Rewrite
$\delta(S_7, \sigma_{CNAME}) \rightarrow (S_1)$

Referral
$\delta(S_8, \sigma_{NS}) \rightarrow (S_2)$

Failover
$\delta(S_9, \sigma_{FAIL}) \rightarrow (S_4)$

② Missing Nameserver Addresses

$\delta(S_2, \sigma_{MISS}) \rightarrow (S_3)$

$\delta(S_3, \sigma_{MISS}) \rightarrow (S_3)$

③ TCP Retransmission

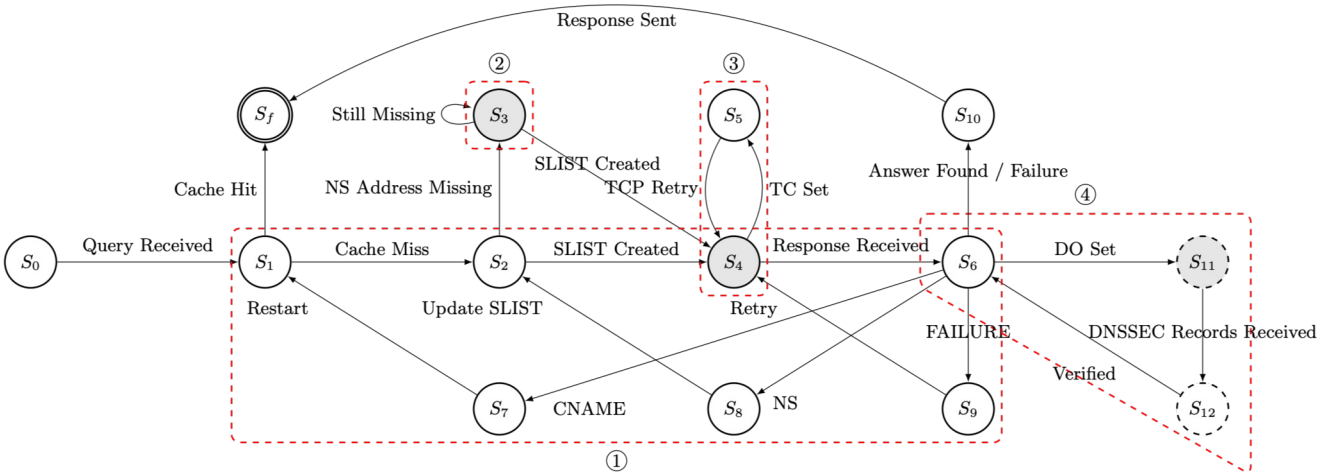
$\delta(S_4, \sigma_{TC}) \rightarrow (S_5)$

④ DNSSEC Validation

$\delta(S_6, \sigma_{DNSSEC}) \rightarrow (S_{11})$

State Transition

$T = (\boldsymbol{S}, \boldsymbol{\Sigma}, \boldsymbol{\delta})$,



① Resolver Top Level Algorithm

Answer
$\delta(S_6, \sigma_{\text{ANSWER}}) \rightarrow (S_{10})$

Rewrite
$\delta(S_7, \sigma_{CNAME}) \rightarrow (S_1)$

Referral
$\delta(S_8, \sigma_{NS}) \rightarrow (S_2)$

Failover
$\delta(S_9, \sigma_{FAIL}) \rightarrow (S_4)$

② Missing Nameserver Addresses

$\delta(S_2, \sigma_{MISS}) \rightarrow (S_3)$

$\delta(S_3, \sigma_{MISS}) \rightarrow (S_3)$

③ TCP Retransmission

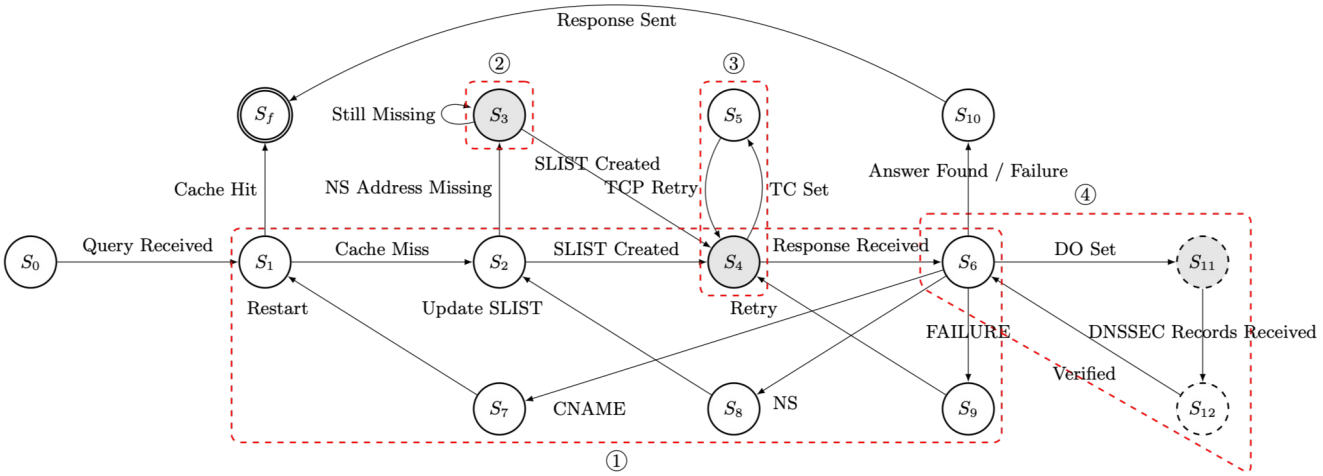
$\delta(S_4, \sigma_{TC}) \rightarrow (S_5)$

④ DNSSEC Validation

$\delta(S_6, \sigma_{DNSSEC}) \rightarrow (S_{11})$

State Transition

$T = (\boldsymbol{S}, \boldsymbol{\Sigma}, \boldsymbol{\delta})$,



① Resolver Top Level Algorithm

Answer
$\delta(S_6, \sigma_{\text{ANSWER}}) \rightarrow (S_{10})$

Rewrite
$\delta(S_7, \sigma_{CNAME}) \rightarrow (S_1)$

Referral
$\delta(S_8, \sigma_{NS}) \rightarrow (S_2)$

Failover
$\delta(S_9, \sigma_{FAIL}) \rightarrow (S_4)$

② Missing Nameserver Addresses

$\delta(S_2, \sigma_{MISS}) \rightarrow (S_3)$

$\delta(S_3, \sigma_{MISS}) \rightarrow (S_3)$

③ TCP Retransmission

$\delta(S_4, \sigma_{TC}) \rightarrow (S_5)$

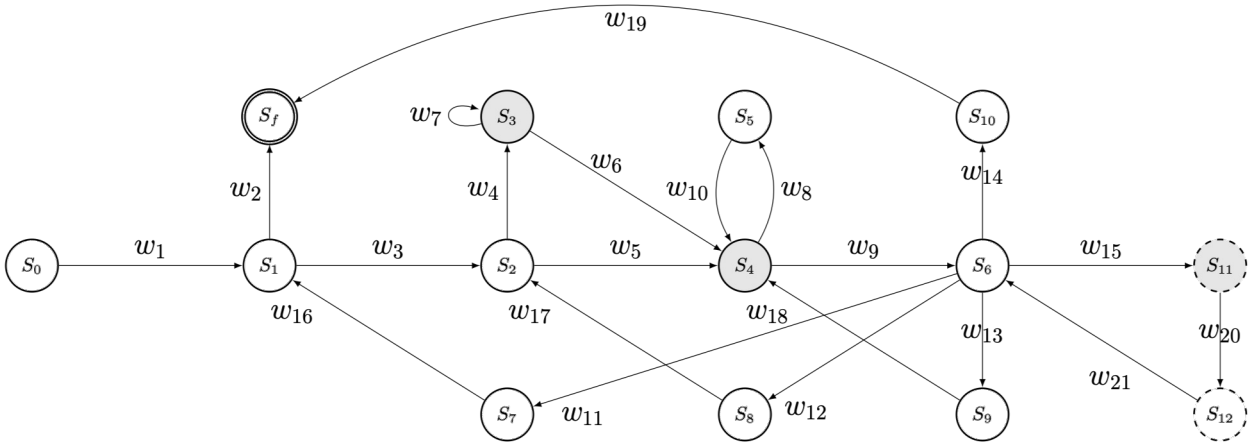
④ DNSSEC Validation

$\delta(S_6, \sigma_{DNSSEC}) \rightarrow (S_{11})$

Modeling state transitions makes resolver behavior tractable.

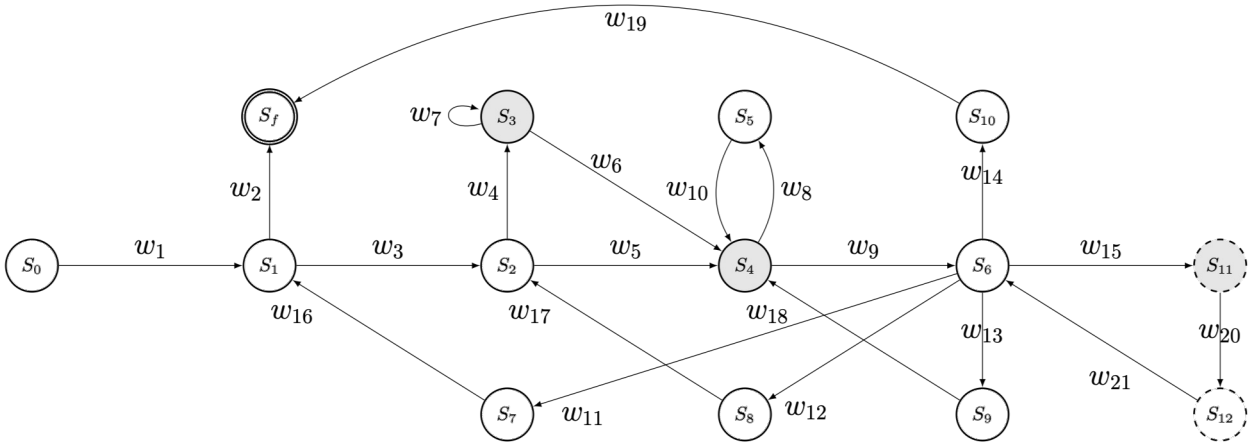
Transition Cost

$T = (S, \Sigma, \delta, \boldsymbol{C})$,



Transition Cost

$T = (S, \Sigma, \delta, \boldsymbol{C})$,



- Outgoing Query Processing Cost

Transport Protocol
TCP vs UDP

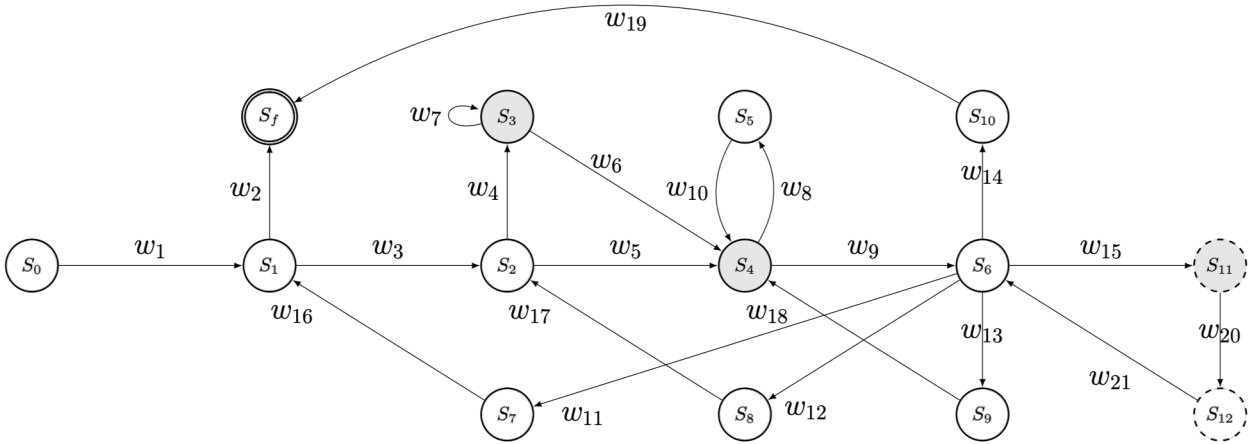
Data Transmission
- RR Size

Response Parsing
- RR Size
- RR Count

Caching
- RR Size
- RR Count
- Cachability

Transition Cost

$T = (S, \Sigma, \delta, \boldsymbol{C})$,



Outgoing Query Processing Cost

Transport Protocol
TCP vs UDP

Data Transmission
- RR Size

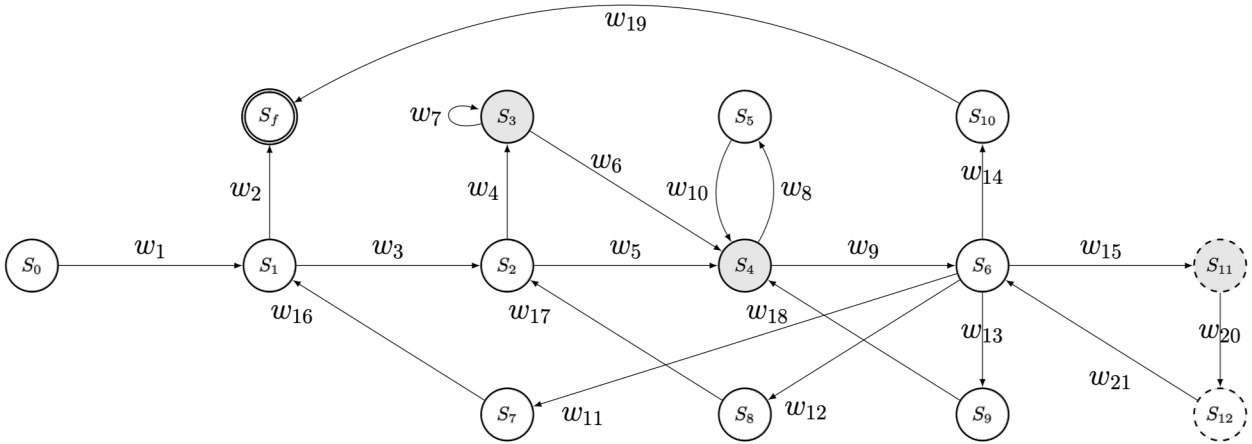
Response Parsing
- RR Size
- RR Count

Caching
- RR Size
- RR Count
- Cachability

DNSSEC Validation Cost
- Algorithm
- Key Size
- RR Count

Transition Cost

$T = (S, \Sigma, \delta, \boldsymbol{C})$,



Outgoing Query Processing Cost

Transport Protocol
TCP vs UDP

Data Transmission
- RR Size

Response Parsing
- RR Size
- RR Count

Caching
- RR Size
- RR Count
- Cachability

DNSSEC Validation Cost

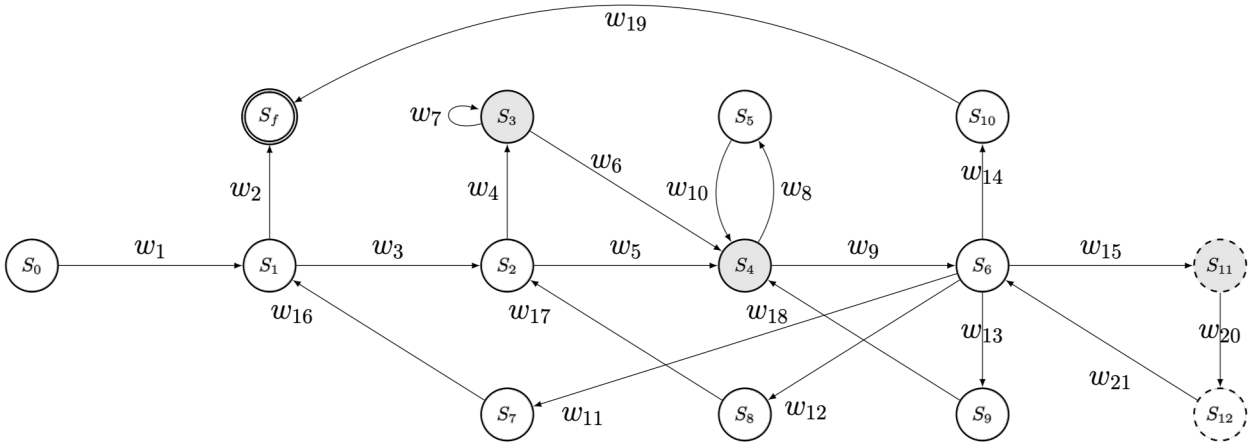
- Algorithm
- Key Size
- RR Count

Fixed Transition Cost

- Constant

Transition Cost

$T = (S, \Sigma, \delta, \boldsymbol{C})$,



Outgoing Query Processing Cost

Transport Protocol
TCP vs UDP

Data Transmission
- RR Size

Response Parsing
- RR Size
- RR Count

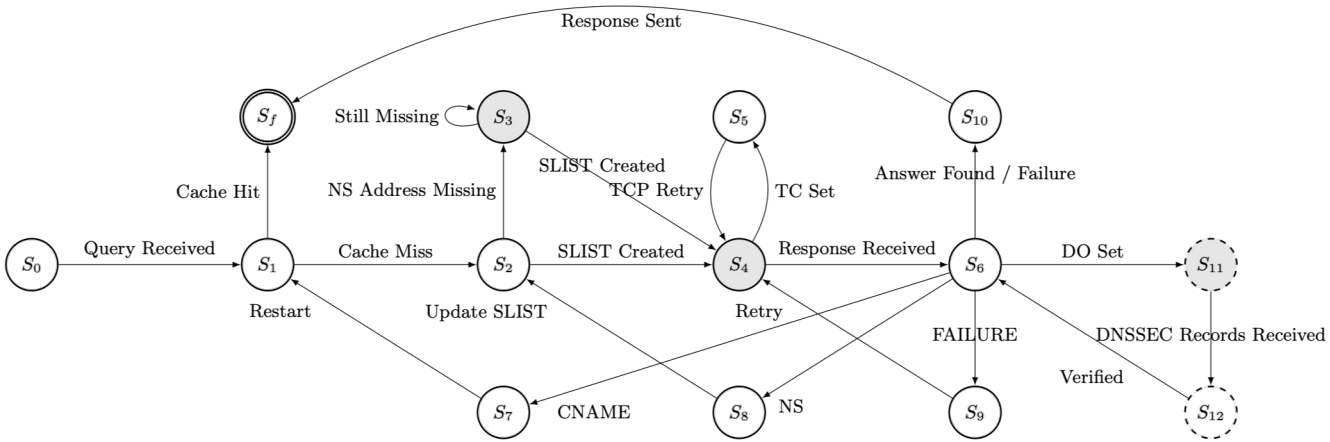
Caching
- RR Size
- RR Count
- Cachability

DNSSEC Validation Cost
- Algorithm
- Key Size
- RR Count

Fixed Transition Cost
- Constant

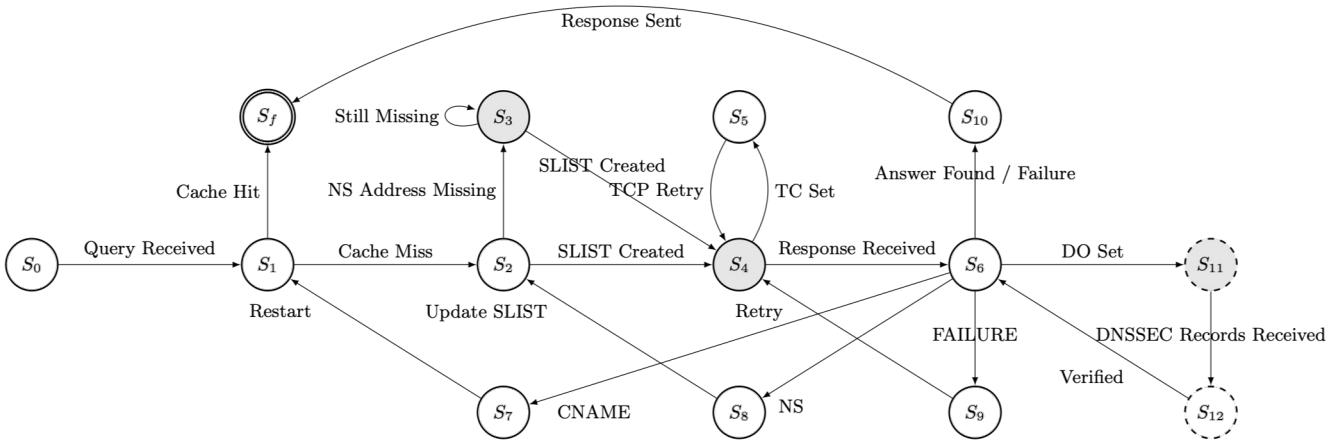
Annotating transitions with cost makes resolver work quantifiable.

Extended Transition System



S	states	stages of resolution	δ	transitions	state transition function
Σ	inputs	server responses	C	cost	resource consumption

Extended Transition System



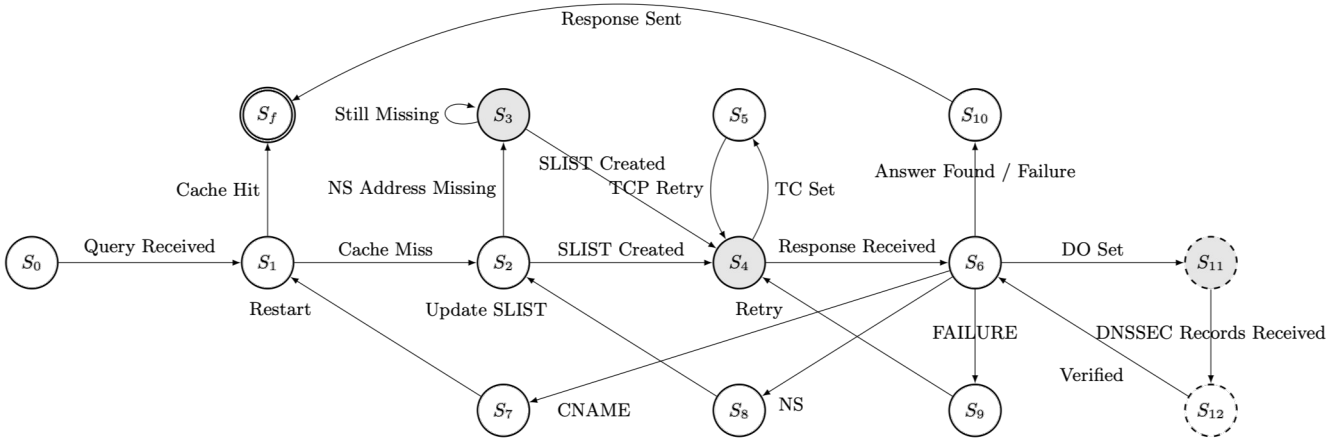
S	states	stages of resolution	δ	transitions	state transition function
Σ	inputs	server responses	C	cost	resource consumption

V **Counters**
track cycle traversal history

L **Constraints**
bound the feasible state space

Extended Transition System

$T = (S, \Sigma, \boldsymbol{V}, \delta, \boldsymbol{L}, C),$



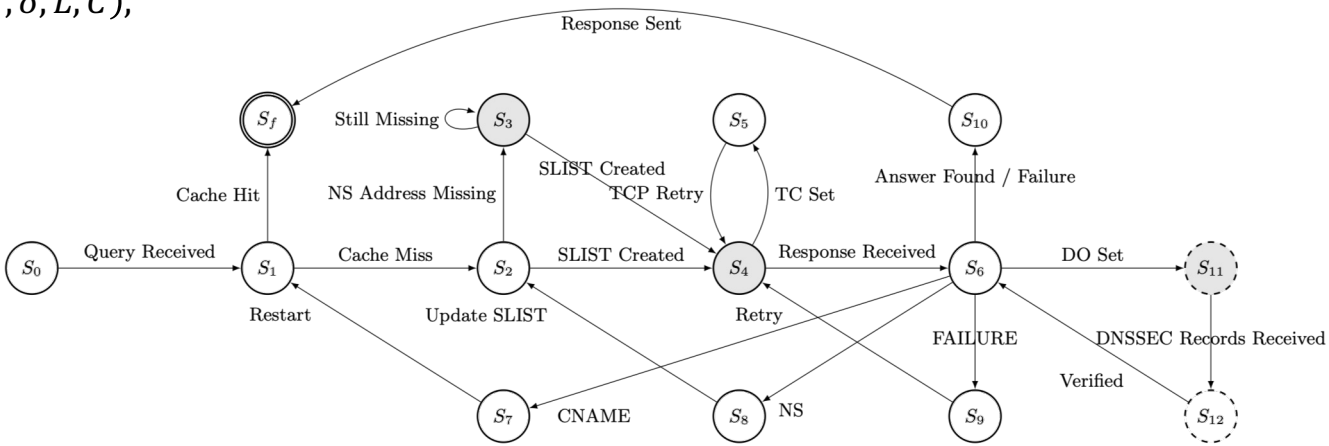
S	states	stages of resolution	δ	transitions	state transition function
Σ	inputs	server responses	C	cost	resource consumption

V **Counters**
track cycle traversal history

L **Constraints**
bound the feasible state space

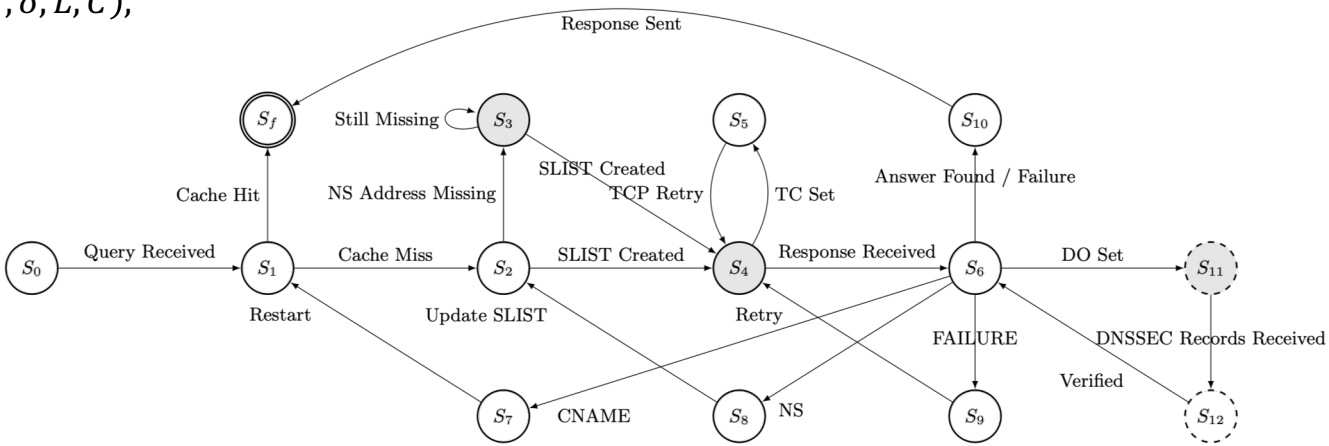
Problem Formulation

$T = (S, \Sigma, V, \delta, L, C)$,



Problem Formulation

$T = (S, \Sigma, V, \delta, L, C)$,



The total work of a resolution is the sum of costs along its state transition path.

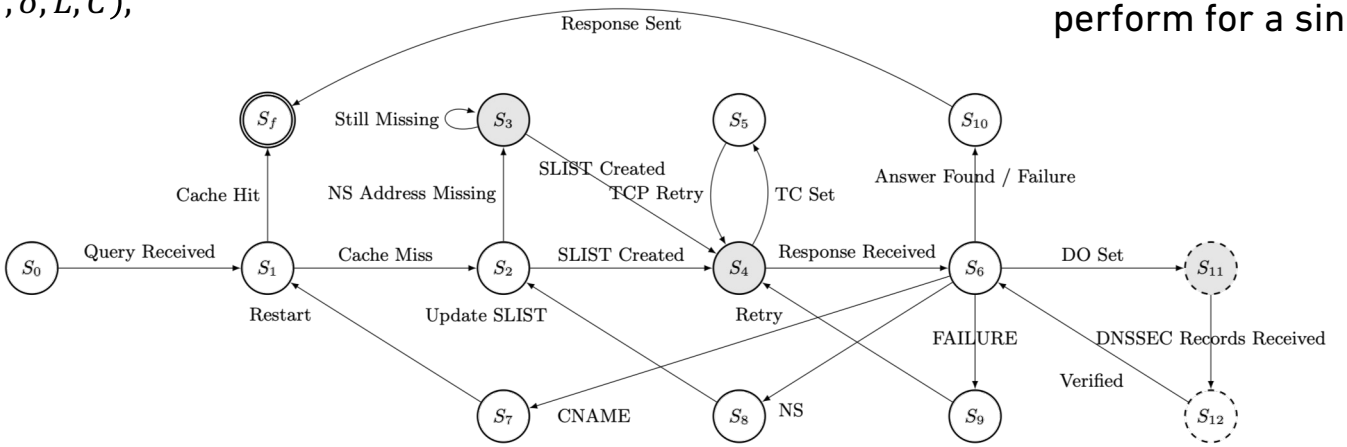
Work

$$W = \sum_{t=1}^{T} w_t$$

Problem Formulation

$T = (S, \Sigma, V, \delta, L, C),$

What is the maximum amount of work a resolver can perform for a single query?

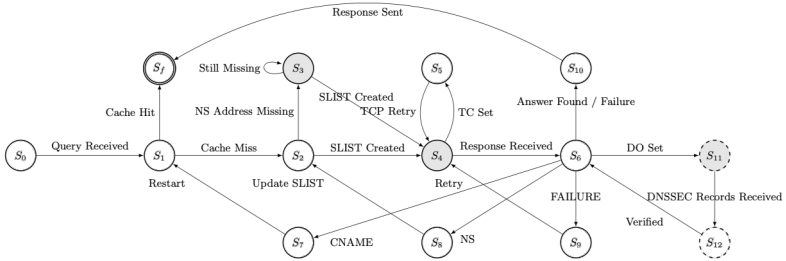


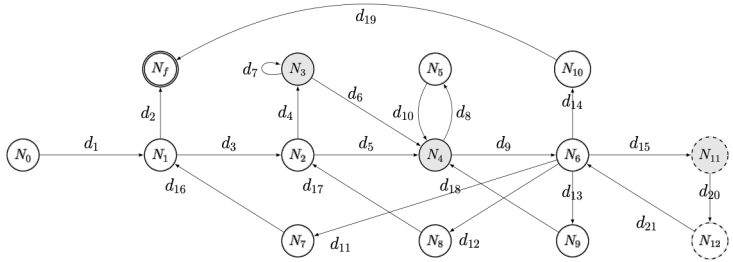
The total work of a resolution is the sum of costs along its state transition path.

Work **Upper Bound**

$$W = \sum_{t=1}^{T} w_t \longrightarrow W^* = \max_{S_0 \to S_f} W$$

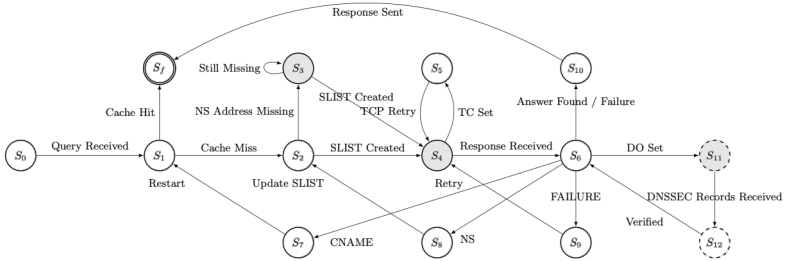
Finding the Worst Case

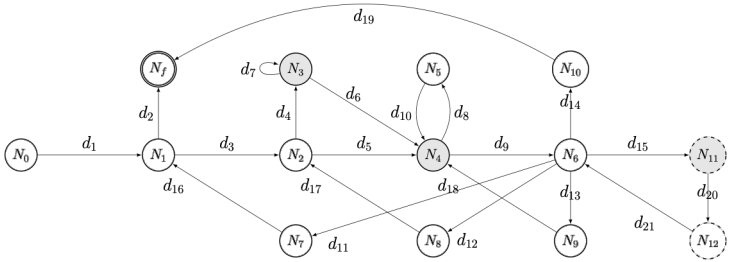




States → Nodes Transitions → Directed Edges Cost → Distance Constraints → Feasibility Bounds

Finding the Worst Case

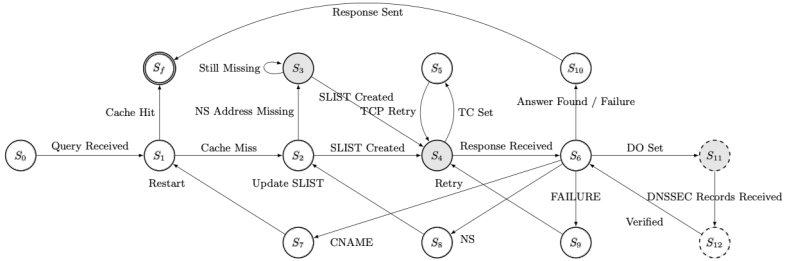


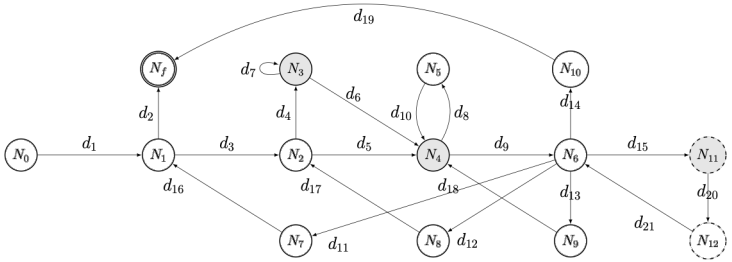


States → Nodes Transitions → Directed Edges Cost → Distance Constraints → Feasibility Bounds

Maximum Work → **Length of the Longest Path**

Finding the Worst Case





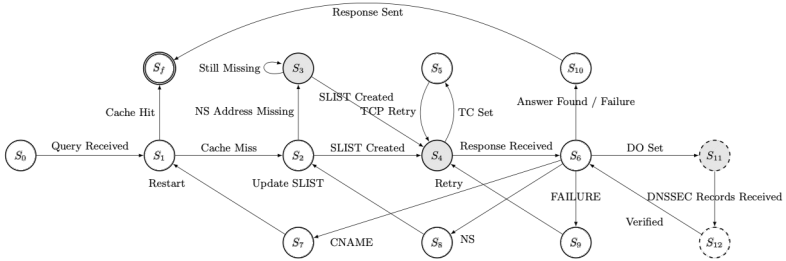
States → Nodes Transitions → Directed Edges Cost → Distance Constraints → Feasibility Bounds

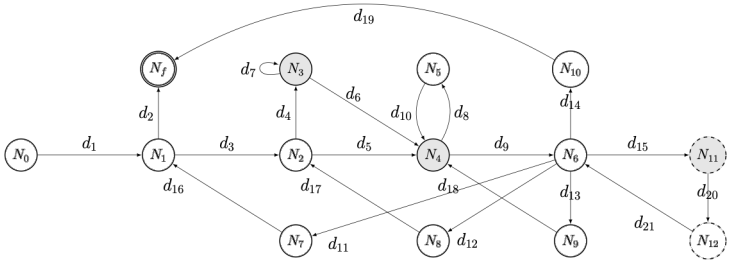
Maximum Work → **Length of the Longest Path**

Constrained Longest Path Search

- Priority search — favor high-cost paths
- Constraint checking — prune infeasible branches at every step
- Termination guarantee — bounded by L

Finding the Worst Case





States → Nodes Transitions → Directed Edges Cost → Distance Constraints → Feasibility Bounds

Maximum Work → **Length of the Longest Path**

W, π*: $S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow \ldots \rightarrow S_f$*
— provably optimal traces

Constrained Longest Path Search

- Priority search — favor high-cost paths
- Constraint checking — prune infeasible branches at every step
- Termination guarantee — bounded by L

Correctness Guarantee

- State Completeness
- Additivity
- Boundedness/Termination

Instantiating Resolution Traces

```
> Init:
     S0 → S1 → S2

> for r = 1 to 11:
     > for d = 1 to 32:
          S2 → S4 → S5 → S4 → S6
          S6 → S11 → S12 → S6
          S6 → S8 → S2
          S2 → S4 → S5 → S4 → S6
          S6 → S11 → S12 → S6
          S6 → S7 → S1 → S2

> for d = 1 to 31:
     S2 → S4 → S5 → S4 → S6
     S6 → S11 → S12 → S6
     S6 → S8 → S2

> Final Answer:
     S2 → S4 → S5 → S4 → S6
     S6 → S11 → S12 → S6
     S6 → S10 → Sf
```

Instantiating Resolution Traces

```
> Init:
     S0 → S1 → S2

> for r = 1 to 11:
     > for d = 1 to 32:
          S2 → S4 → S5 → S4 → S6
          S6 → S11 → S12 → S6
          S6 → S8 → S2
          S2 → S4 → S5 → S4 → S6
          S6 → S11 → S12 → S6
          S6 → S7 → S1 → S2

> for d = 1 to 31:
     S2 → S4 → S5 → S4 → S6
     S6 → S11 → S12 → S6
     S6 → S8 → S2

> Final Answer:
     S2 → S4 → S5 → S4 → S6
     S6 → S11 → S12 → S6
     S6 → S10 → Sf
```

A formal model and algorithm are necessary but insufficient
— standard nameservers cannot serve the traces they produce.

We need a system that can…

- Generates responses dynamically
- Session-aware across queries
- Full control over content, timing, ordering

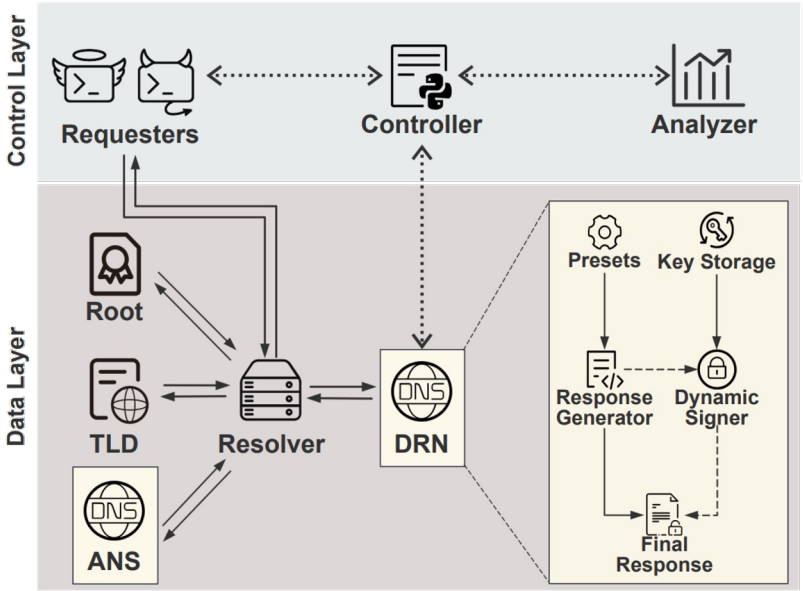
Instantiating Resolution Traces

```
> Init:
     S0 → S1 → S2

> for r = 1 to 11:
     > for d = 1 to 32:
          S2 → S4 → S5 → S4 → S6
          S6 → S11 → S12 → S6
          S6 → S8 → S2
          S2 → S4 → S5 → S4 → S6
          S6 → S11 → S12 → S6
          S6 → S7 → S1 → S2

> for d = 1 to 31:
     S2 → S4 → S5 → S4 → S6
     S6 → S11 → S12 → S6
     S6 → S8 → S2

> Final Answer:
     S2 → S4 → S5 → S4 → S6
     S6 → S11 → S12 → S6
     S6 → S10 → Sf
```

A formal model and algorithm are necessary but insufficient
— standard nameservers cannot serve the traces they produce.

We need a system that can…

- Generates responses dynamically
- Session-aware across queries
- Full control over content, timing, ordering



rProfiler

From Constraints to Measured Resolver Work

On State Transitions

Parameters	**BIND** 9.21.1	**Unbound** 1.21.1	**PowerDNS** 5.1.3
max_restart	11	11	10
max_depth	32	128	50
max_fetch	5	6	10
max_referral	7	4	16
cname_reset	True	True	False
max_outgoing_query	384	128	50
tcp_retry_included	False	False	True
dnssec_included	False	False	True

From Constraints to Measured Resolver Work

On State Transitions

Parameters	BIND 9.21.1	Unbound 1.21.1	PowerDNS 5.1.3
max_restart	11	11	10
max_depth	32	128	50
max_fetch	5	6	10
max_referral	7	4	16
cname_reset	True	True	False
max_outgoing_query	384	128	50
tcp_retry_included	False	False	True
dnssec_included	False	False	True

On Transition Cost

Response	Type	Bind9 9.21.1	Unbound 1.21.1	PowerDNS 5.1.3
Answer	QTYPE	✓	✓	✓, ↔, ↓
	RRSIG	○	○	○
Authoritative	NS	✓	✓	✓, ↔, ↓
	DS	○	○	○
	RRSIG	○	○	○
Additional	A	✓	✓	○
	AAAA	✓	✓	✓, ↔, ↓
	RRSIG	○	○	○

From Constraints to Measured Resolver Work

On State Transitions

Parameters	BIND 9.21.1	Unbound 1.21.1	PowerDNS 5.1.3
max_restart	11	11	10
max_depth	32	128	50
max_fetch	5	6	10
max_referral	7	4	16
cname_reset	True	True	False
max_outgoing_query	384	128	50
tcp_retry_included	False	False	True
dnssec_included	False	False	True

On Transition Cost

Response	Type	Bind9 9.21.1	Unbound 1.21.1	PowerDNS 5.1.3
Answer	QTYPE	✓	✓	✓, ↔, ↓
	RRSIG	○	○	○
Authoritative	NS	✓	✓	✓, ↔, ↓
	DS	○	○	○
	RRSIG	○	○	○
Additional	A	✓	✓	○
	AAAA	✓	✓	✓, ↔, ↓
	RRSIG	○	○	○

Some constraints are implicit — only discoverable through probing.

From Constraints to Measured Resolver Work

On State Transitions

Parameters	BIND 9.21.1	Unbound 1.21.1	PowerDNS 5.1.3
max_restart	11	11	10
max_depth	32	128	50
max_fetch	5	6	10
max_referral	7	4	16
cname_reset	True	True	False
max_outgoing_query	384	128	50
tcp_retry_included	False	False	True
dnssec_included	False	False	True

On Transition Cost

Response	Type	Bind9 9.21.1	Unbound 1.21.1	PowerDNS 5.1.3
Answer	QTYPE	✓	✓	✓, ↔, ↓
	RRSIG	○	○	○
Authoritative	NS	✓	✓	✓, ↔, ↓
	DS	○	○	○
	RRSIG	○	○	○
Additional	A	✓	✓	○
	AAAA	✓	✓	✓, ↔, ↓
	RRSIG	○	○	○

Some constraints are implicit — only discoverable through probing.

Sample Traces

 # TRX = 1536

> Init:
> $S_0 \to S_1 \to S_2$
> for r = 1 to 11:
> > for d = 1 to 32:
> > $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> > $S_6 \to S_{11} \to S_{12} \to S_6$
> > $S_6 \to S_8 \to S_2$
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_7 \to S_1 \to S_2$
> for d = 1 to 31:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_8 \to S_2$
> Final Answer:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_{10} \to S_f$

(a) Bind9

 # TRX = 512

> Init:
> $S_0 \to S_1 \to S_2$
> for d = 1 to 127:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_8 \to S_2$
> Final Answer:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_{10} \to S_f$

(b) Unbound

 # TRX = 96 (with IPv6)

> Init:
> $S_0 \to S_1 \to S_2$
> for d = 1 to 15:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_8 \to S_2$
> Final Answer:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_{10} \to S_f$

(c) PowerDNS

From Constraints to Measured Resolver Work

On State Transitions

Parameters	BIND 9.21.1	Unbound 1.21.1	PowerDNS 5.1.3
max_restart	11	11	10
max_depth	32	128	50
max_fetch	5	6	10
max_referral	7	4	16
cname_reset	True	True	False
max_outgoing_query	384	128	50
tcp_retry_included	False	False	True
dnssec_included	False	False	True

On Transition Cost

Response	Type	Bind9 9.21.1	Unbound 1.21.1	PowerDNS 5.1.3
Answer	QTYPE	✓	✓	✓, ↔, ↓
	RRSIG	○	○	○
Authoritative	NS	✓	✓	✓, ↔, ↓
	DS	○	○	○
	RRSIG	○	○	○
Additional	A	✓	✓	○
	AAAA	✓	✓	✓, ↔, ↓
	RRSIG	○	○	○

Some constraints are implicit — only discoverable through probing.

Sample Traces

 # TRX = 1536

> Init:
> $S_0 \to S_1 \to S_2$
> for r = 1 to 11:
> for d = 1 to 32:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_8 \to S_2$
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_7 \to S_1 \to S_2$
> for d = 1 to 31:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_8 \to S_2$
> Final Answer:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_{10} \to S_f$

(a) Bind9

 # TRX = 512

> Init:
> $S_0 \to S_1 \to S_2$
> for d = 1 to 127:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_8 \to S_2$
> Final Answer:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_{10} \to S_f$

(b) Unbound

 # TRX = 96 (with IPv6)

> Init:
> $S_0 \to S_1 \to S_2$
> for d = 1 to 15:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_8 \to S_2$
> Final Answer:
> $S_2 \to S_4 \to S_5 \to S_4 \to S_6$
> $S_6 \to S_{11} \to S_{12} \to S_6$
> $S_6 \to S_{10} \to S_f$

(c) PowerDNS

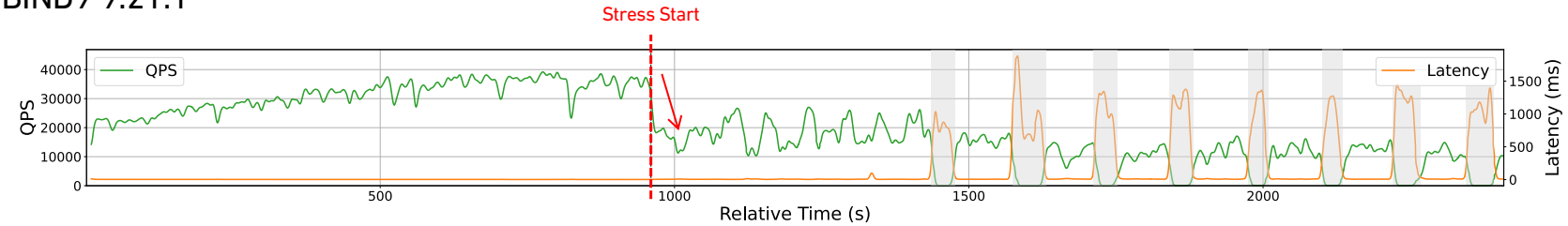
One single query can trigger...

Resolver	Transactions	Packets	Volume	Validation	Cache
BIND	1459 / 1536	12369	38.84 M	377	35.57 M
Unbound	493 / 512	5381	13.12 M	132	14.28 M
PowerDNS†	80 / 96	363	2.13 M	30	3.84 M

30 QPS Is More Than Enough To Exhaust Resolvers

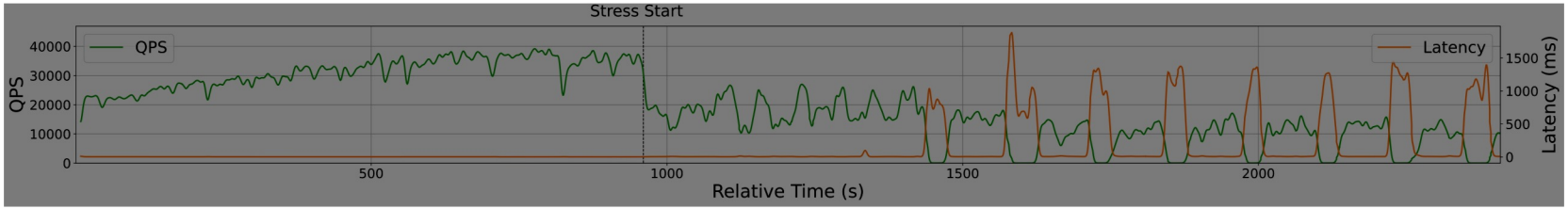
30 QPS Is More Than Enough To Exhaust Resolvers

BIND9 9.21.1

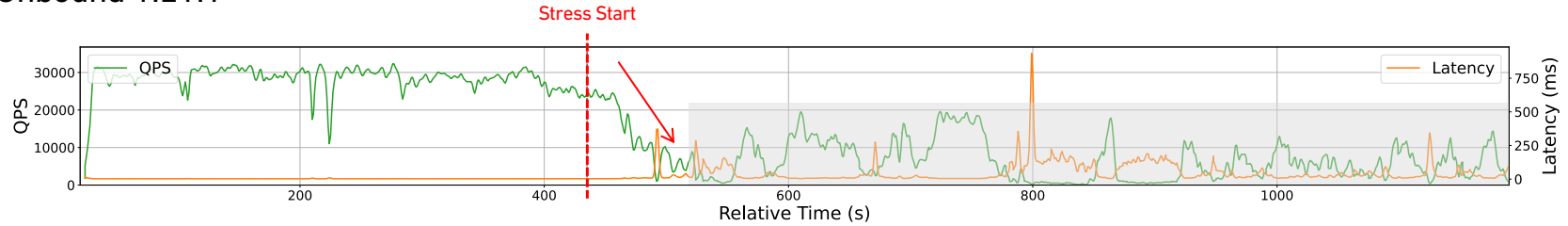


30 QPS Is More Than Enough To Exhaust Resolvers

BIND9 9.21.1

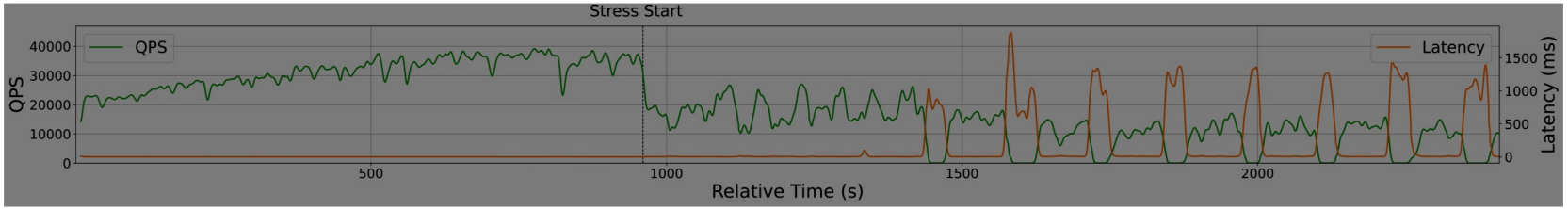


Unbound 1.21.1

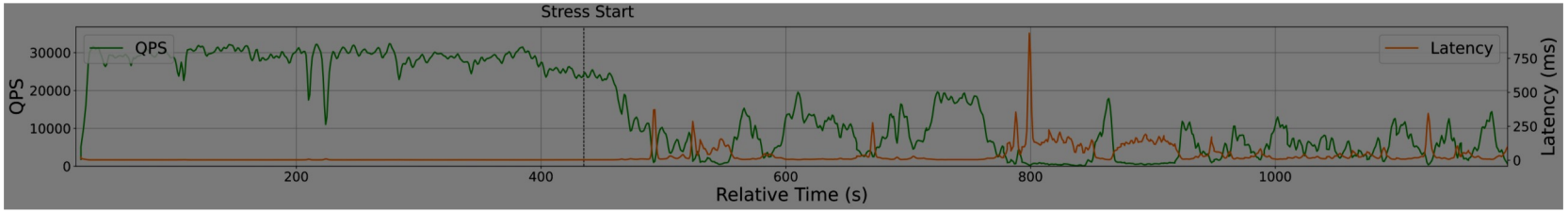


30 QPS Is More Than Enough To Exhaust Resolvers

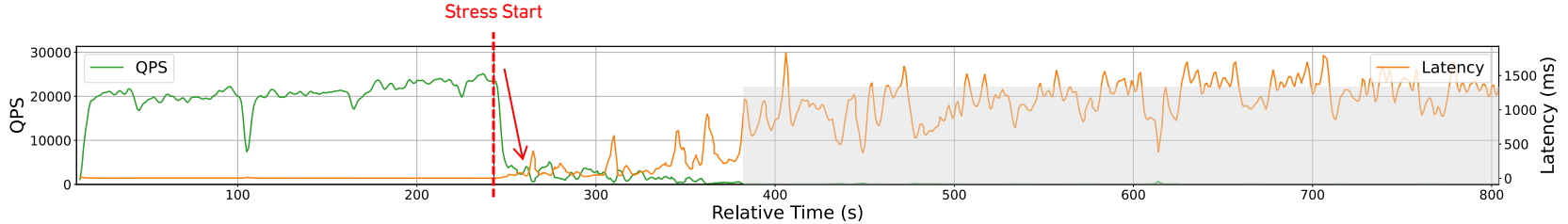
BIND9 9.21.1



Unbound 1.21.1

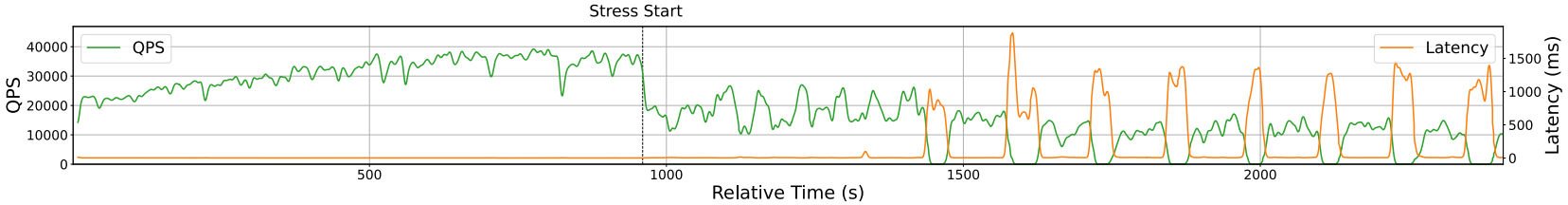


PowerDNS 5.1.3

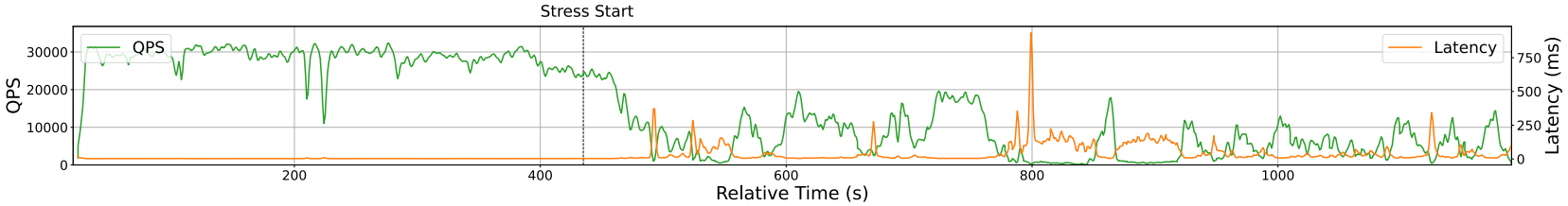


30 QPS Is More Than Enough To Exhaust Resolvers

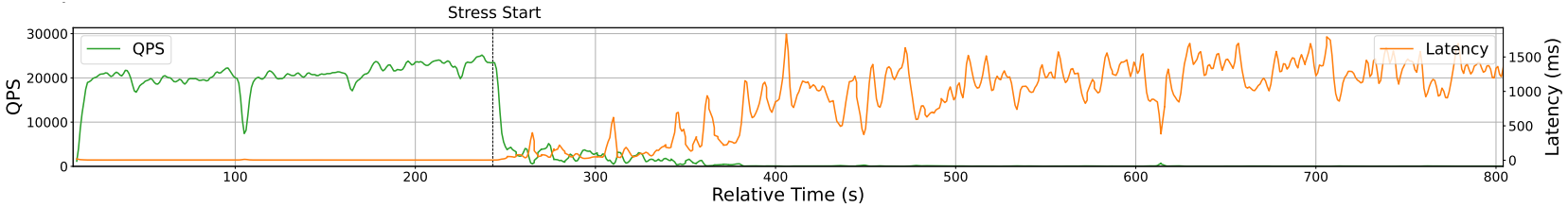
BIND9 9.21.1



Unbound 1.21.1



PowerDNS 5.1.3



Takeaway

Resolver work can be and should be formally defined and computed

> The recommended priorities for the resolver designer are:
>
> 1. Bound the amount of work (packets sent, parallel processes started)…
>
> – RFC1034

Takeaway

Resolver work can be and should be formally defined and computed

> The recommended priorities for the resolver designer are:
>
> 1. Bound the amount of work (packets sent, parallel processes started)…
>
> – RFC1034

Resolvers remain vulnerable even with all mitigations in place

Takeaway

Resolver work can be and should be formally defined and computed

> The recommended priorities for the resolver designer are:
>
> 1. Bound the amount of work (packets sent, parallel processes started)…
>
> – RFC1034

Resolvers remain vulnerable even with all mitigations in place

Performance degradation is multi-faceted

Takeaway

Resolver work can be and should be formally defined and computed

> The recommended priorities for the resolver designer are:
>
> 1. Bound the amount of work (packets sent, parallel processes started)…
>
> – RFC1034

Resolvers remain vulnerable even with all mitigations in place

Performance degradation is multi-faceted

A principled foundation for future resolver design

Towards End-to-End Availability Guarantees

- **Understand resolver work**
- RFC Disambiguation
- Proactive resource scheduling

Takeaway

Resolver work can be and should be formally defined and computed

> The recommended priorities for the resolver designer are:
>
> 1. Bound the amount of work (packets sent, parallel processes started)…
>
> – RFC1034

Resolvers remain vulnerable even with all mitigations in place

Performance degradation is multi-faceted

A principled foundation for future resolver design

Towards End-to-End Availability Guarantees

- **Understand resolver work**
- RFC Disambiguation
- Proactive resource scheduling

SCAN ME

Resolve the Unresolved: Systematic Work Profiling for DNS Resolvers.

Contact: liwen.xu@inf.ethz.ch