Subverting BIND's SRTT Algorithm

Roee Hay
IBM

Jonathan Kalechstein
Technion

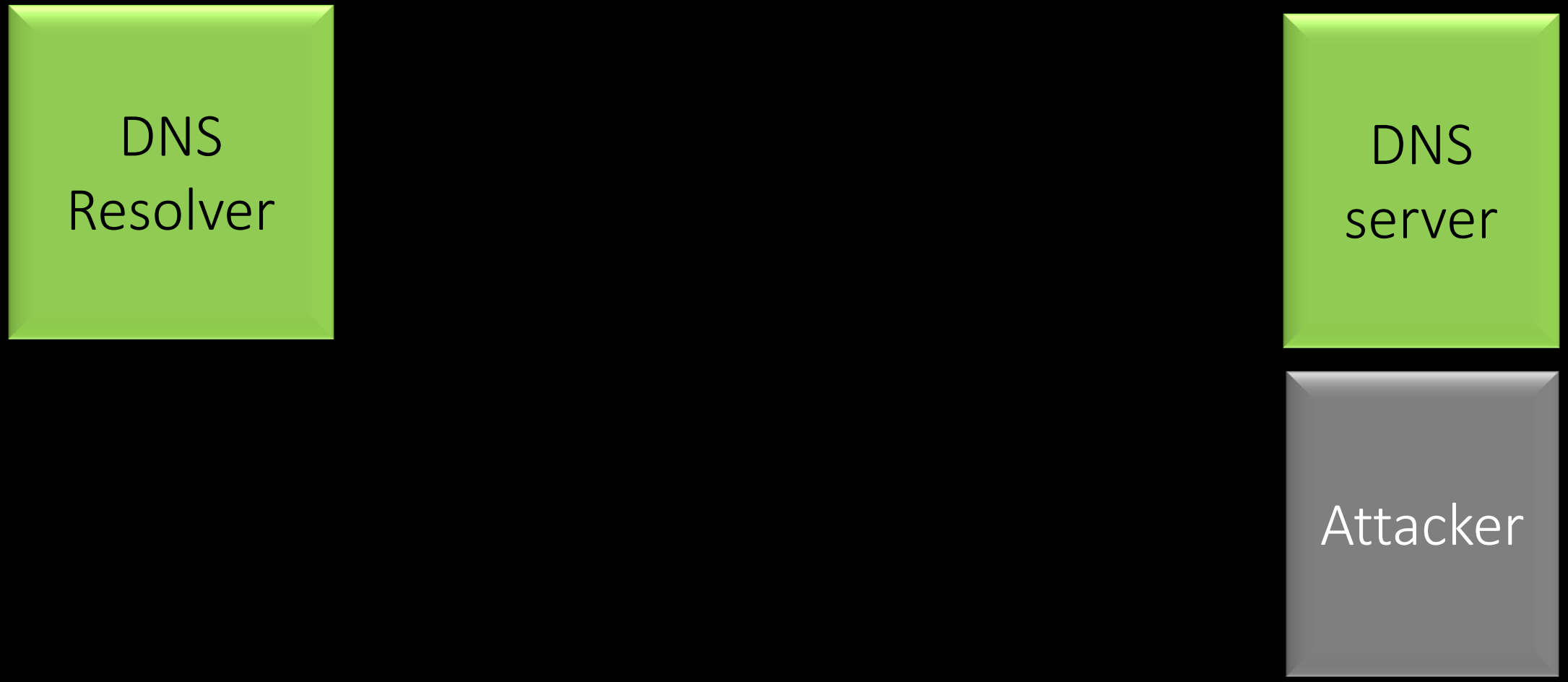
Gabi Nakibly
National EW Research
& Simulation Center

Agenda

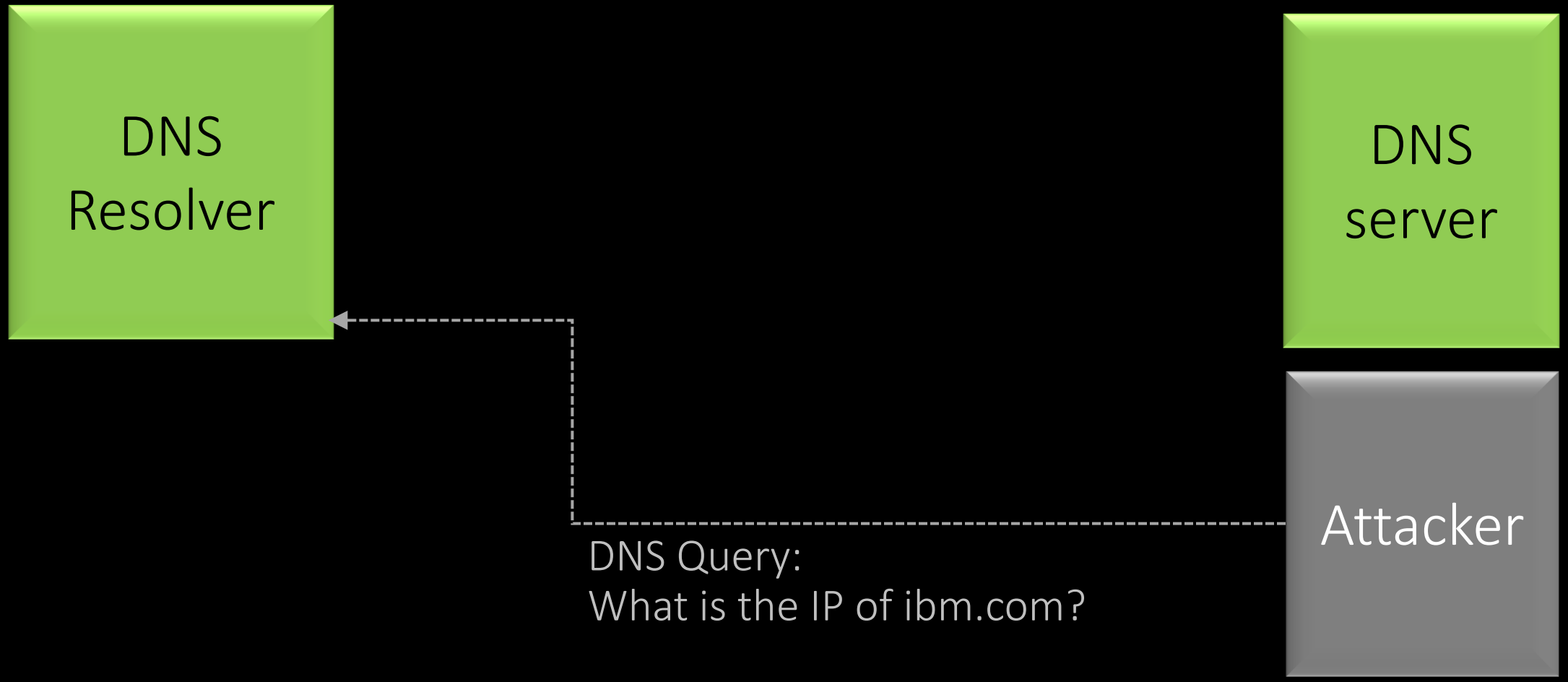
- Off-path (blind) DNS cache poisoning attacks
- BIND’s name server (NS) selection algorithm and previous attacks
- The new attack

Off-Path DNS Cache Poisoning

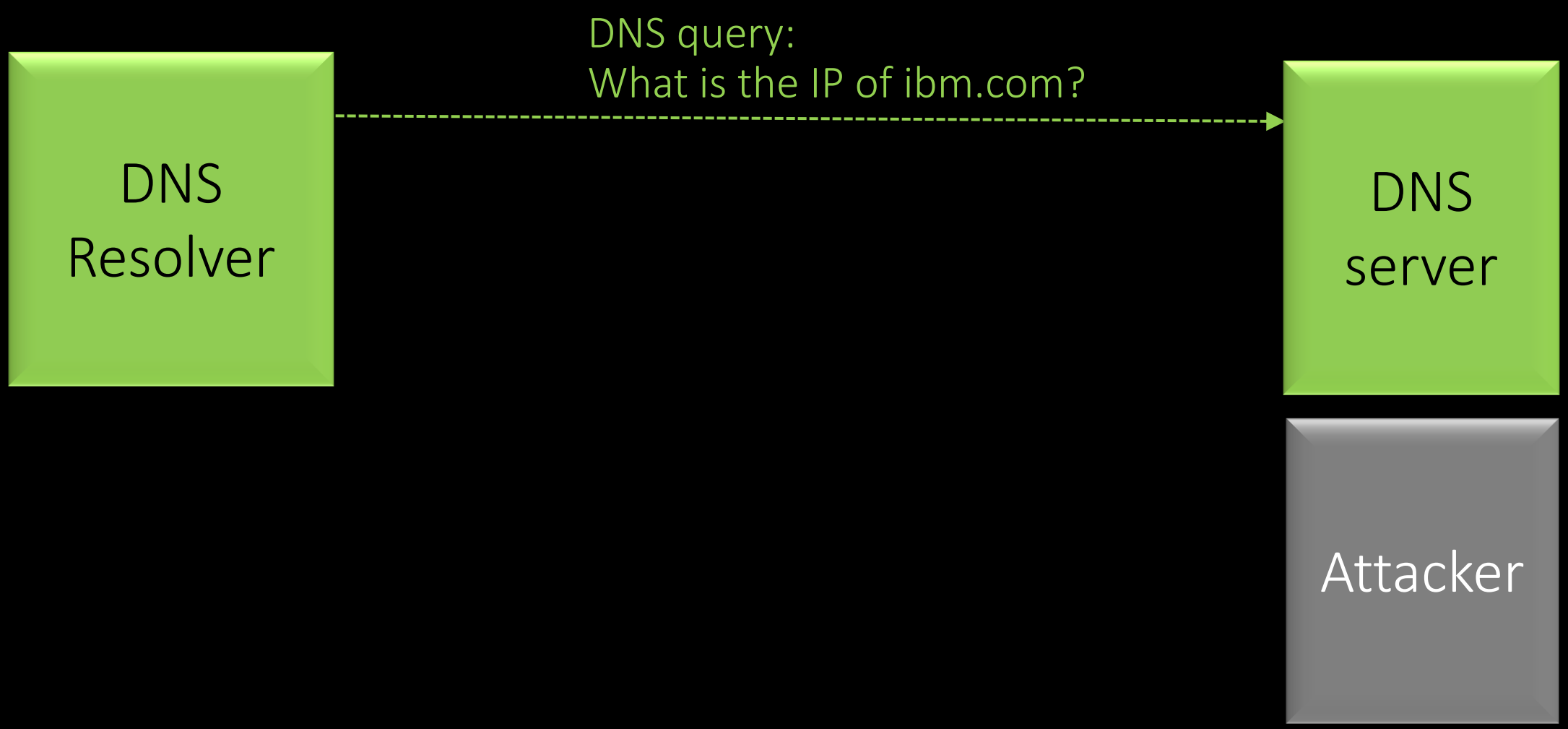
A Trivial Scenario



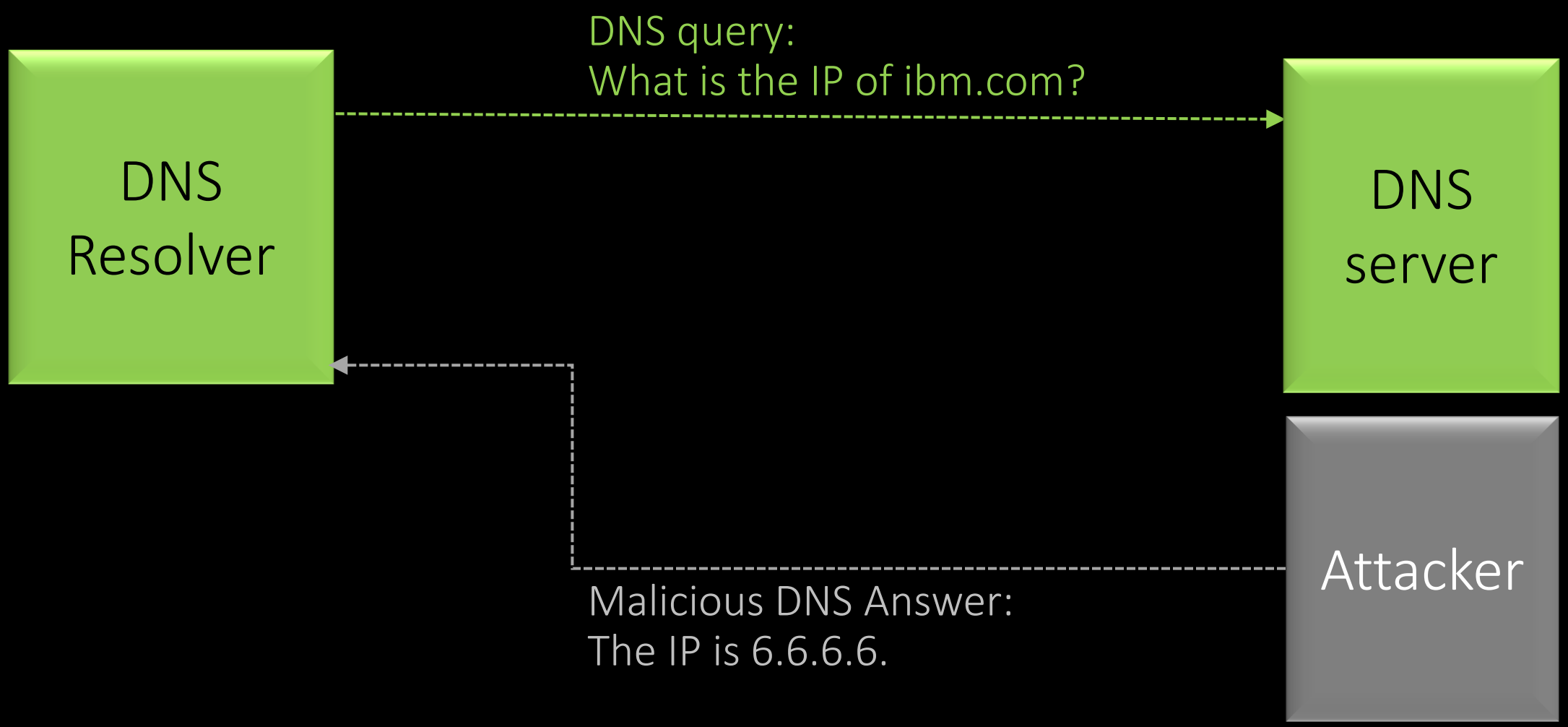
A Trivial Scenario



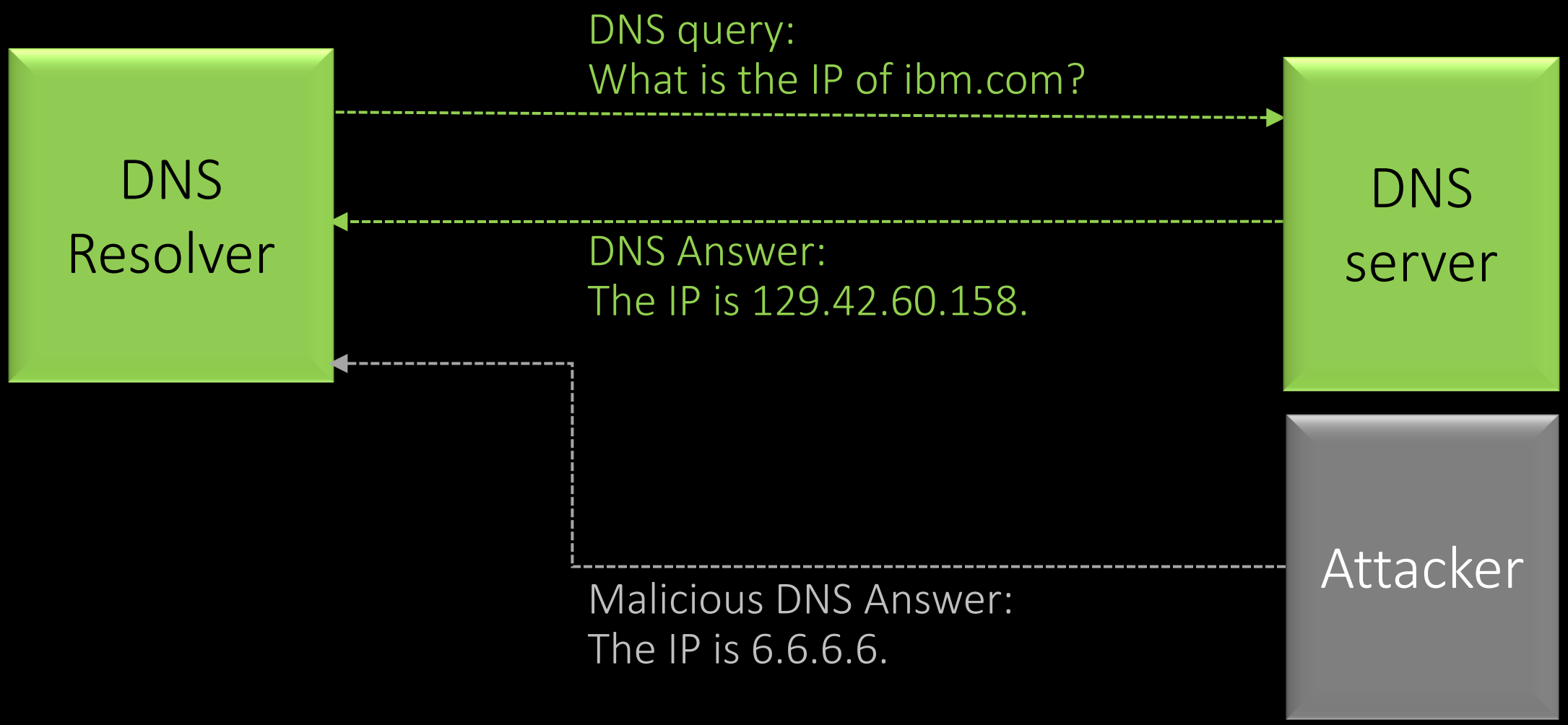
A Trivial Scenario



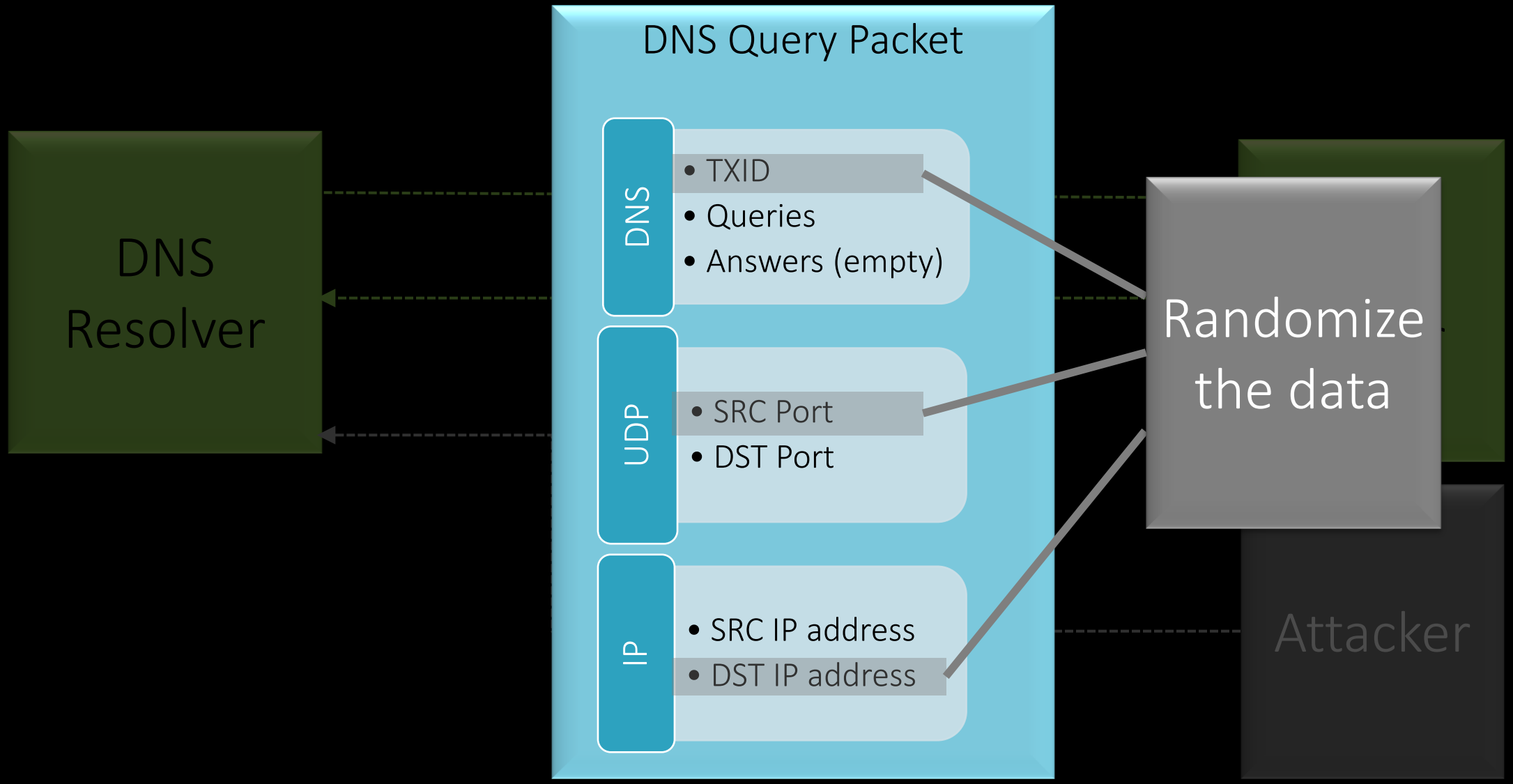
A Trivial Scenario



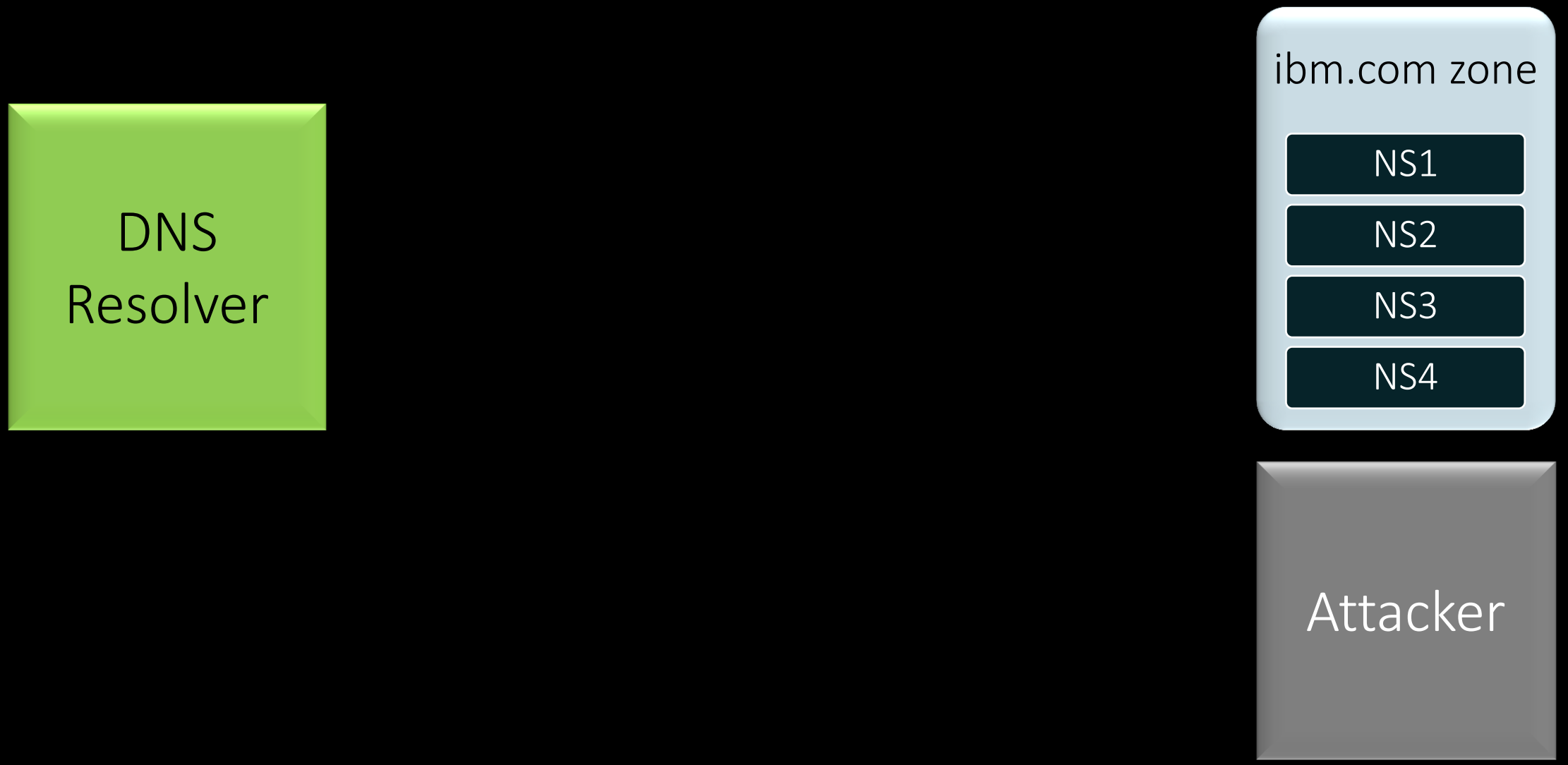
A Trivial Scenario



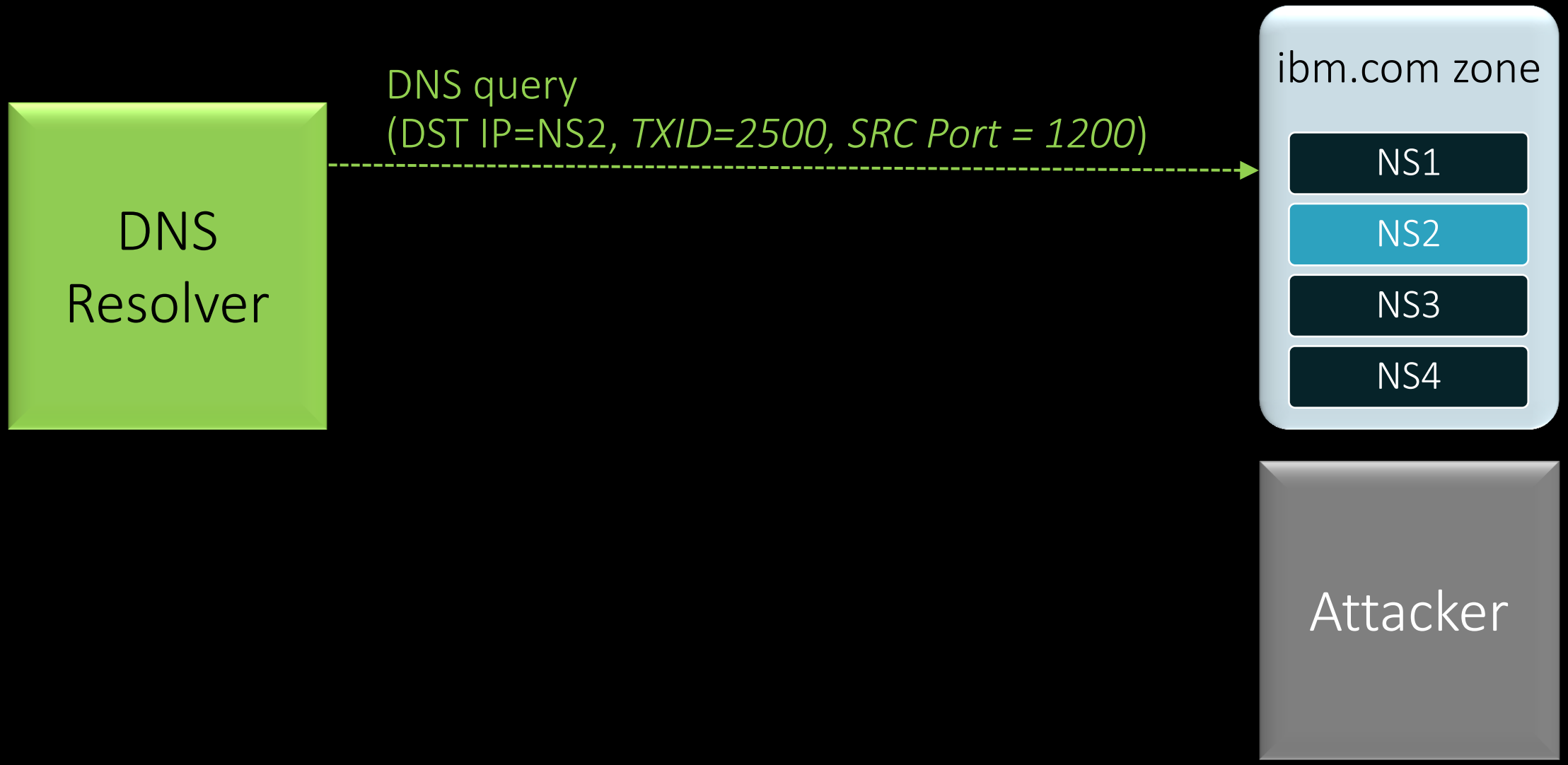
Common Protection Against Off-Path Attacks



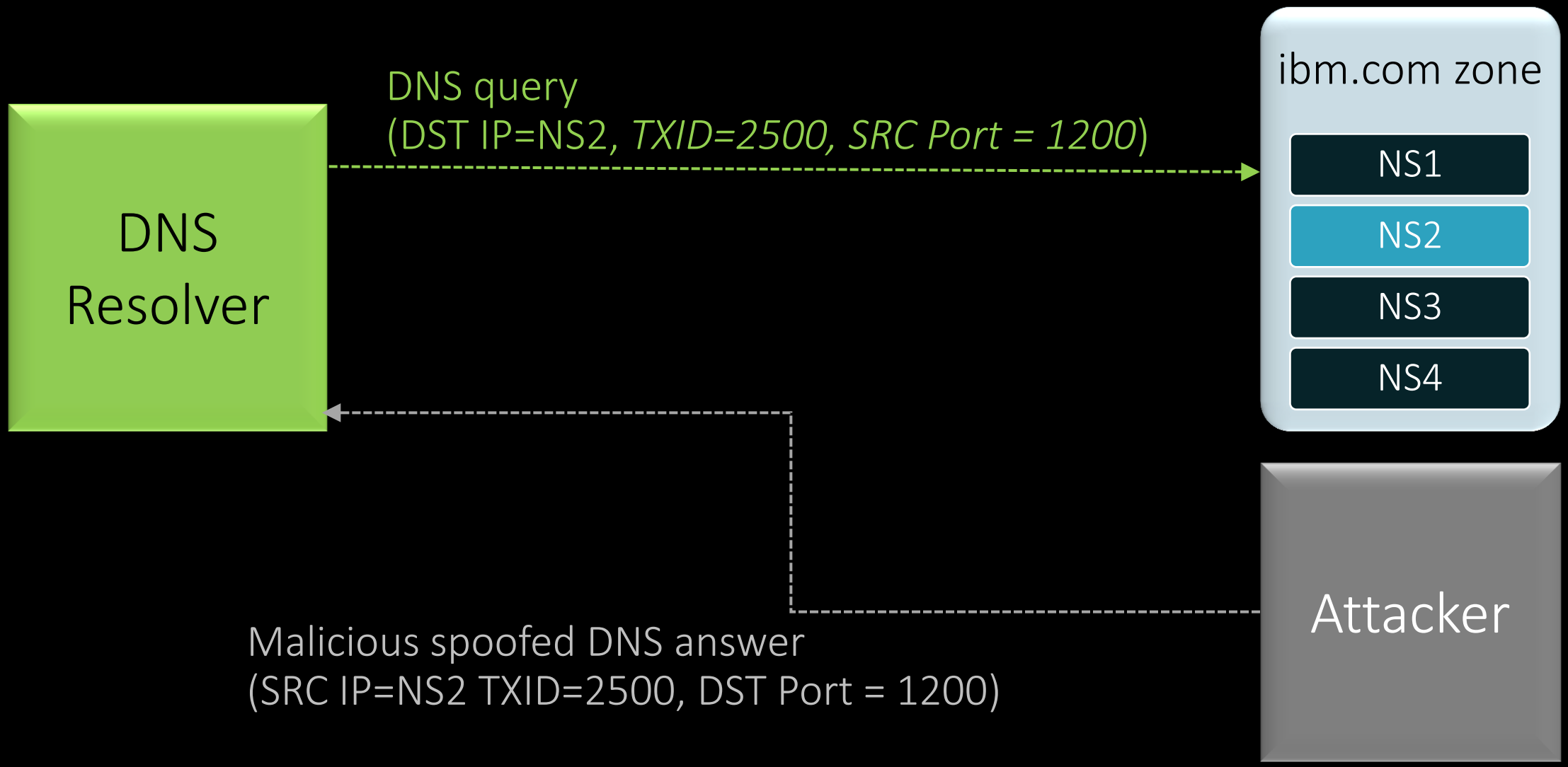
Off-path DNS Poisoning: An Actual Attack



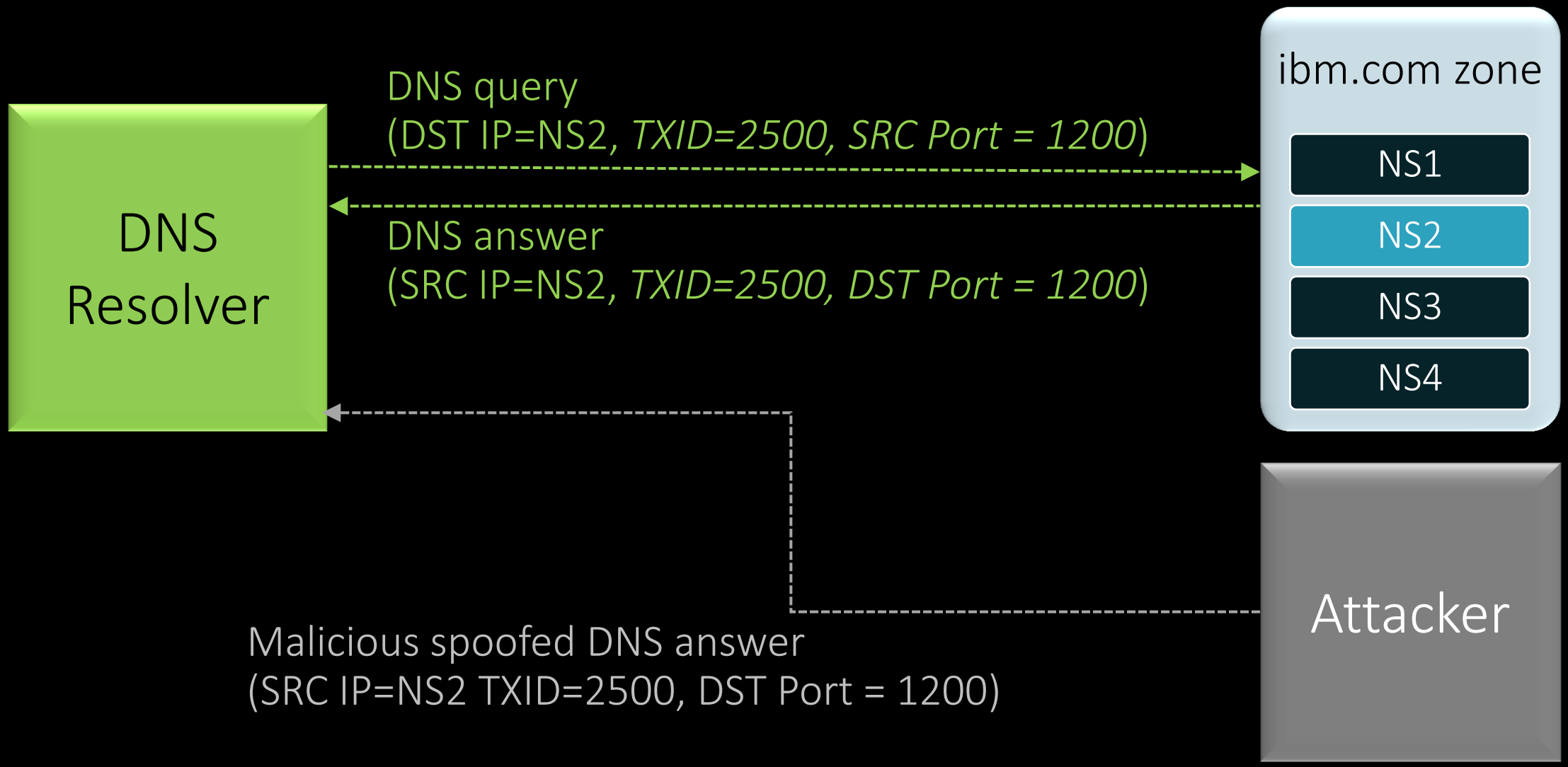
Off-path DNS Poisoning: An Actual Attack



Off-path DNS Poisoning: An Actual Attack

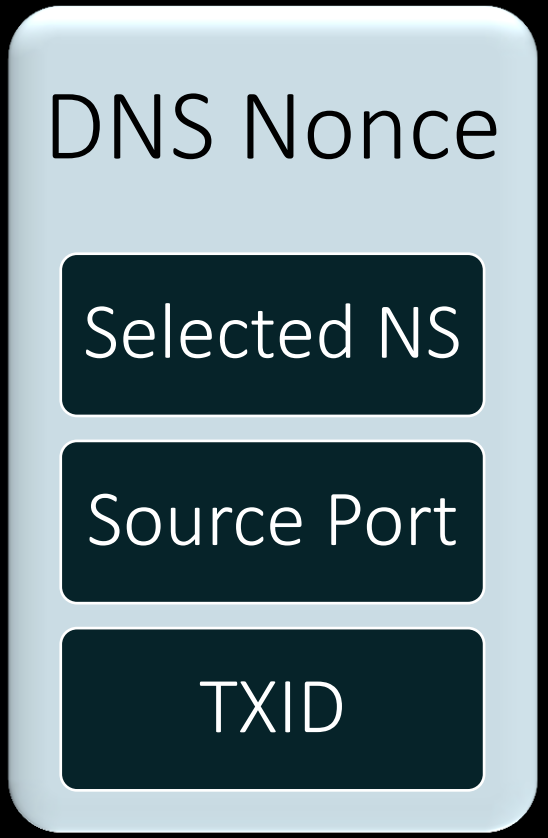


Off-path DNS Poisoning: An Actual Attack



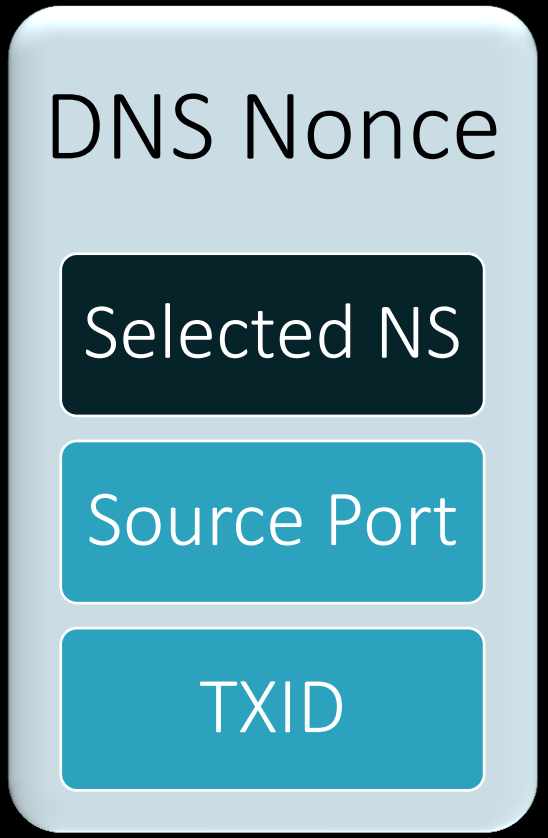
Motivation

- The security of the DNS transaction directly depends on the nonce's randomness.



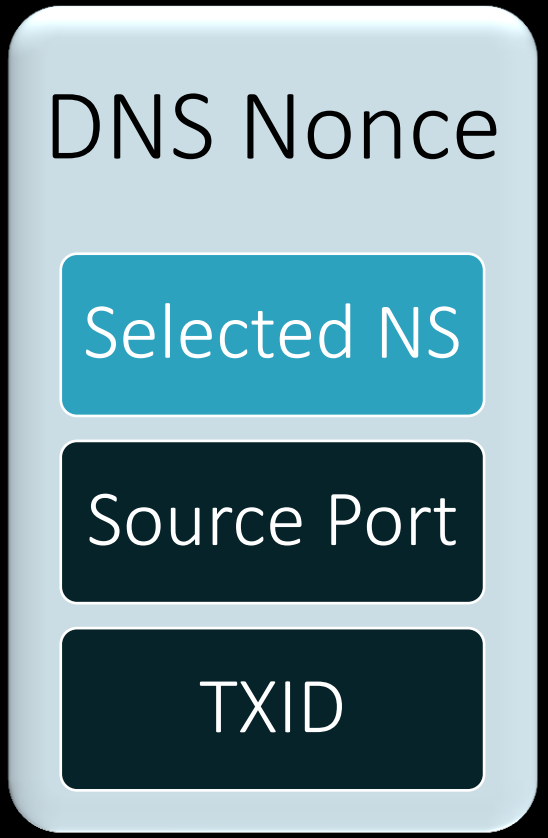
Motivation

- The security of the DNS transaction directly depends on the nonce's randomness.
- The source port and TXID are well studied.

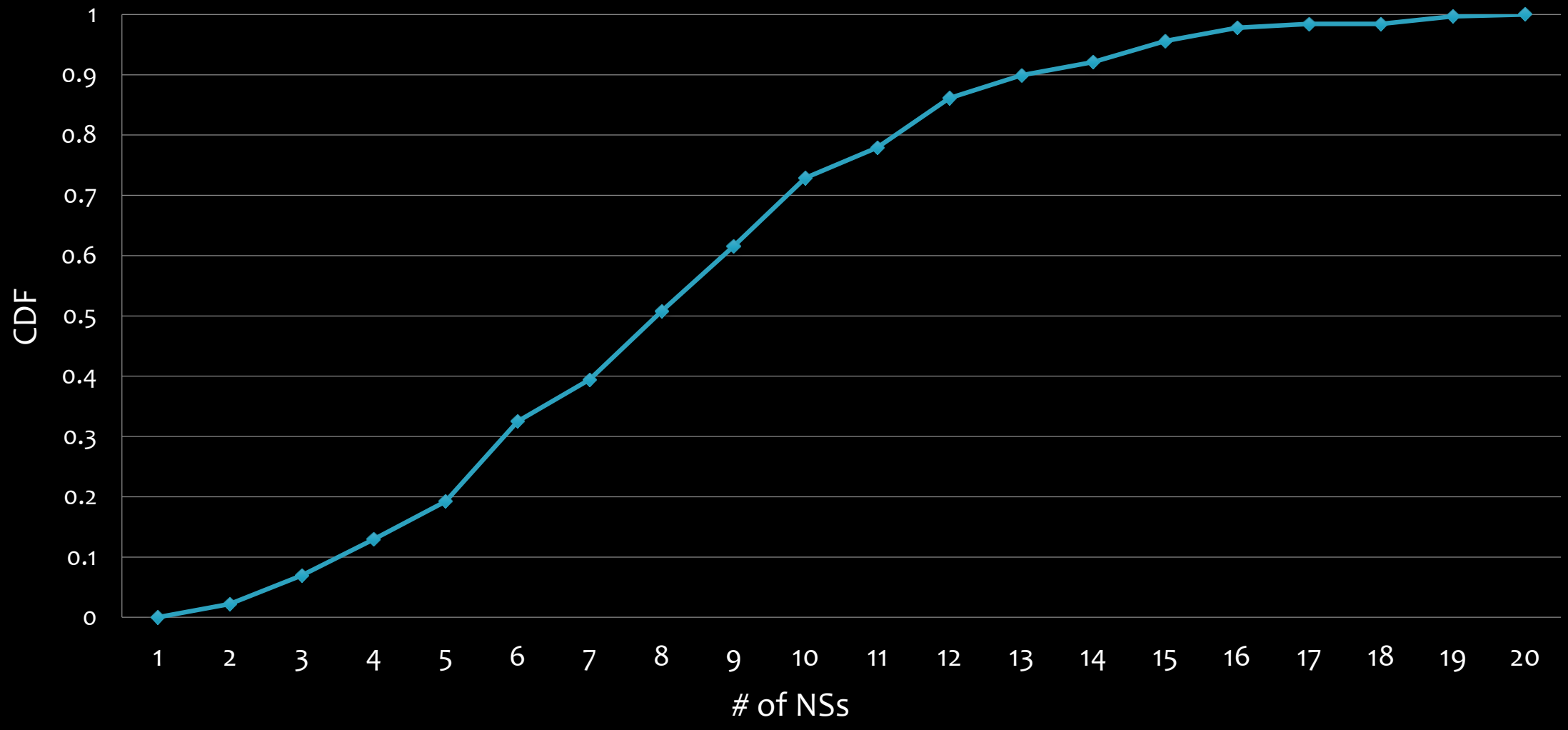


Motivation

- The security of the DNS transaction directly depends on the nonce's randomness.
- The source port and TXID are well studied.
- We try to tackle the NS selection.
  - Derandomizing only the NS selection does not make an off-path attack feasible. It makes existing attacks more efficient, i.e. faster.
  - It enables on-path (Man-in-the-Middle) attacks if the attacker is on one path between the resolver and the NS, but not on another.



CDF of the # of NSs (Top-Level Domains only)



* Data parsed out of the root's zone file: http://www.iana.org/domains/root/files

BIND’s NS Selection and Attacks

BIND's NS Selection: The Smoothed RTT Algorithm

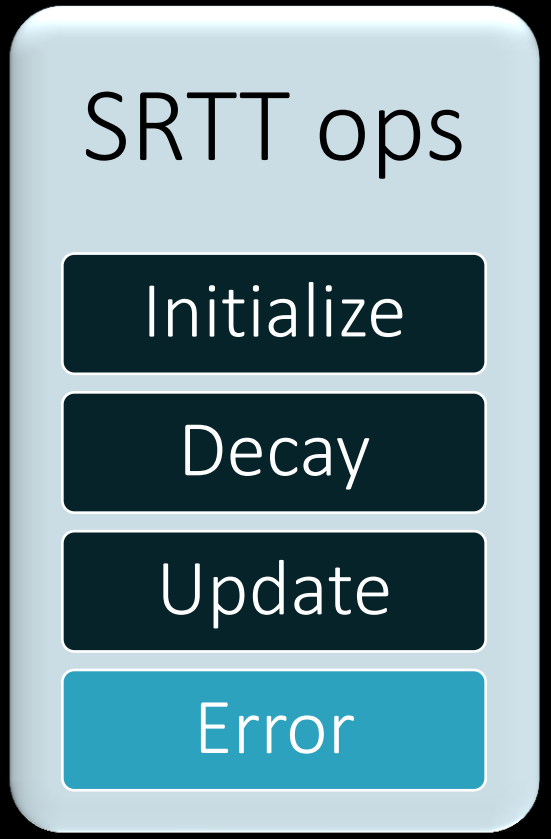
- **Goal**. Choose the most responsive (by Round-Trip Time) NS.
- **Problem**. RTT changes frequently.
- **Data structure**. A moving average for each NS IP.
- **Operations.**
  - Initialize $SRTT \in [1,32]\ \mu s$
  - Update $SRTT = 0.7 \cdot SRTT_{old} + 0.3 \cdot RTT$
  - Decay $SRTT = 0.98 \cdot SRTT_{old}$
  - Error $SRTT = \min(SRTT_{old} + 200ms, 1s)$
- **Cache.** A map keyed *only* by NS IPs is maintained.
- **Selection.** Candidate NS with lowest SRTT value is queried first.

The SRTT Algorithm: A Potential Vulnerability

- The NS selection is derandomized if we can control the SRTT value of the candidates.
- Either by:
  - Increasing all candidates but one
  - Decreasing the victim NS.
- Since the cache stores all NSs together, maybe we can control it externally by a malicious NS?

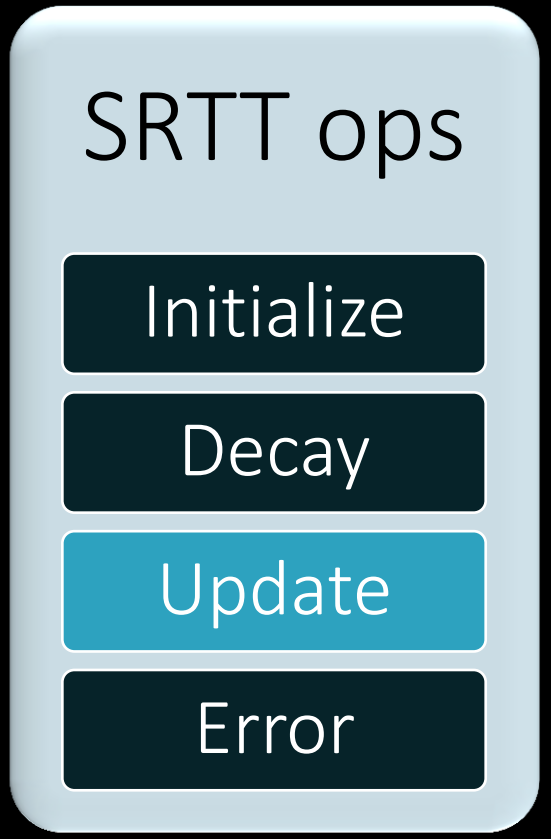
Previous Work and our Contribution

- [Herzberg & Shulman, 2012] increases the SRTT of all candidates NSs but one by abusing fragmented IP packets.



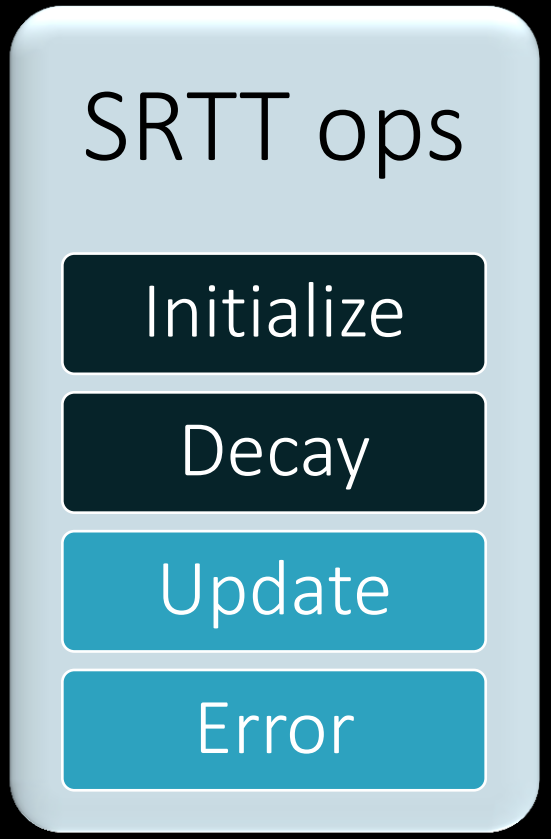
Previous Work and our Contribution

- [Herzberg & Shulman, 2012] increases the SRTT of all candidates NSs but one by abusing fragmented IP packets.
- [Petr, 2009] decreases the SRTT of the victim NS by fast spoofed responses.



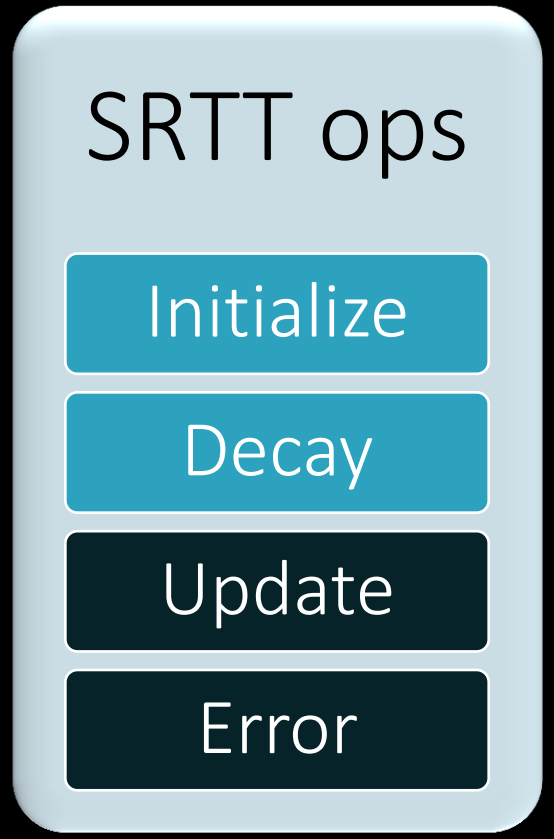
Previous Work and our Contribution

- [Herzberg & Shulman, 2012] increases the SRTT of all candidates NSs but one by abusing fragmented IP packets.
- [Petr, 2009] decreases the SRTT of the victim NS by fast spoofed responses.
- These attacks are **probabilistic**.



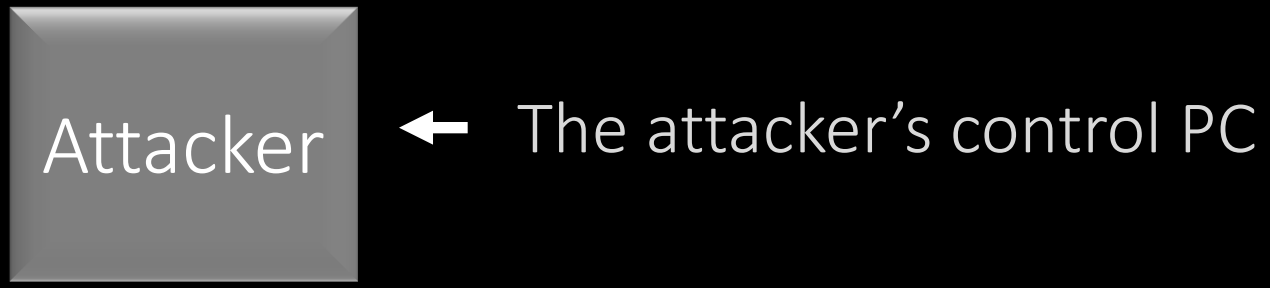
Previous Work and our Contribution

- [Herzberg & Shulman, 2012] increases the SRTT of all candidates NSs but one by abusing fragmented IP packets.
- [Petr, 2009] decreases the SRTT of the victim NS by fast spoofed responses.
- These attacks are **probabilistic**.
- We present a **deterministic** attack against the *Decay* and *Initialize* operations.
  - Another cool feature: The victim NS does not see our attack.



The New Attack

General Setting of the Attack



← The attacker’s control PC

General Setting of the Attack

Attacker ← The attacker's control PC

A_1 ← An attacker's controlled NS.
Authoritative of the a1.foo. domain

General Setting of the Attack

Attacker ← The attacker's control PC

A_1 ← An attacker's controlled NS.
Authoritative of the a1.foo. domain

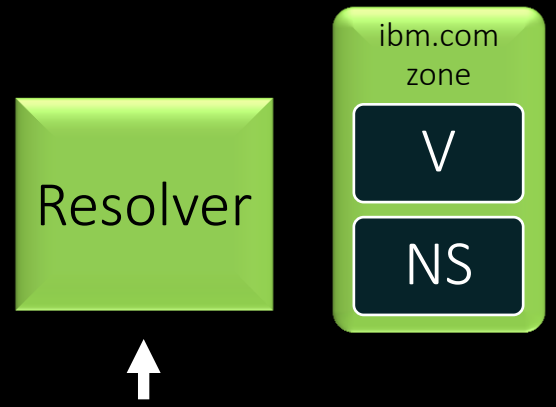
A_2 ← An attacker's controlled NS.
Authoritative of the a2.foo. domain

General Setting of the Attack

Attacker

A_1

A_2



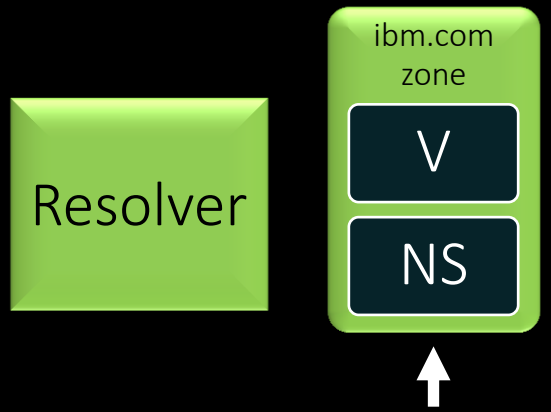
The resolver which we attack

General Setting of the Attack

Attacker

A_1

A_2



We lower the SRTT of V on the resolver, thus force it to be selected when resolving ibm.com.
The target zone can have an arbitrary amount of authoritative name servers.

General Setting of the Attack

Attacker

A_1

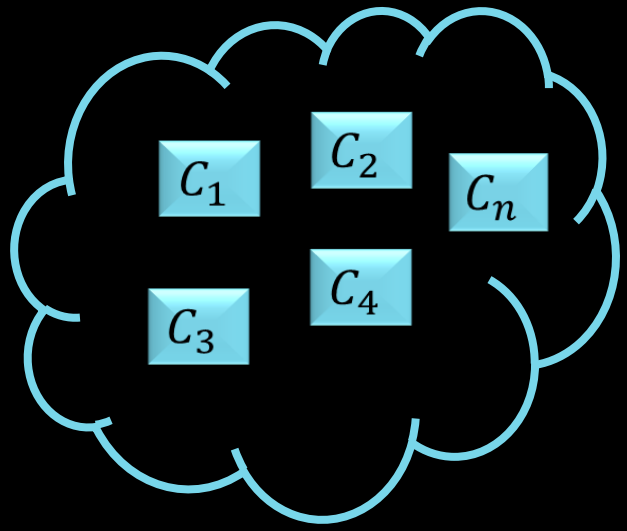
A_2

Resolver

ibm.com zone

V

NS



n non-open NSs which we abuse. Can be anywhere on the Internet. We don't need to control them.

The Attack

Attacker

A_1

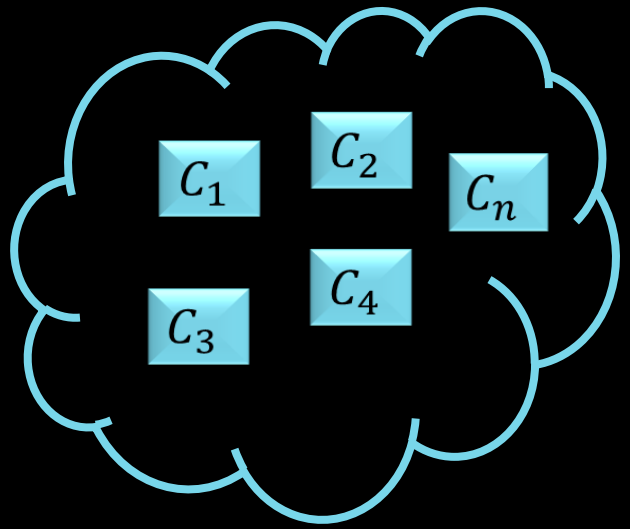
A_2

Resolver

ibm.com zone

V

NS



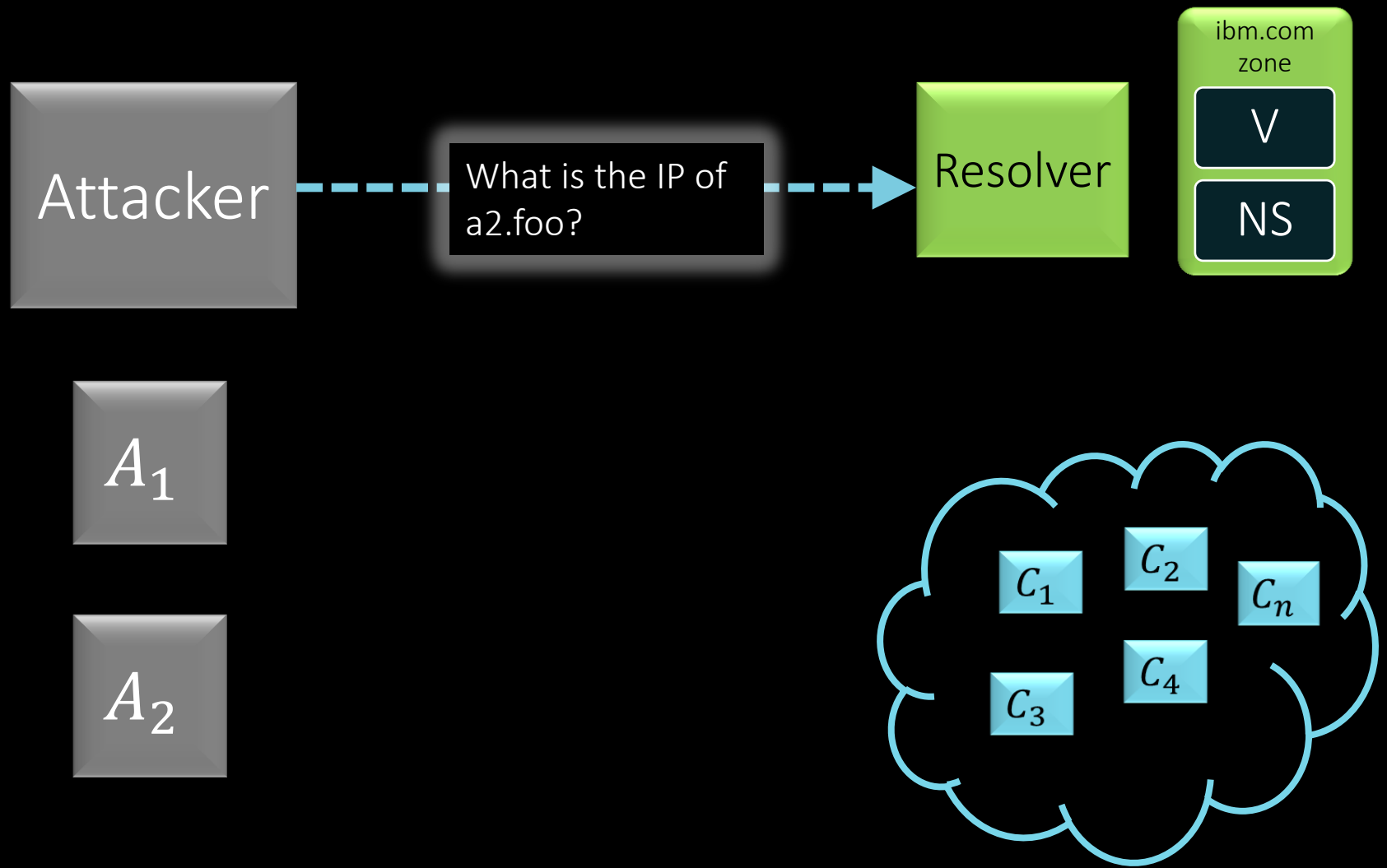
Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



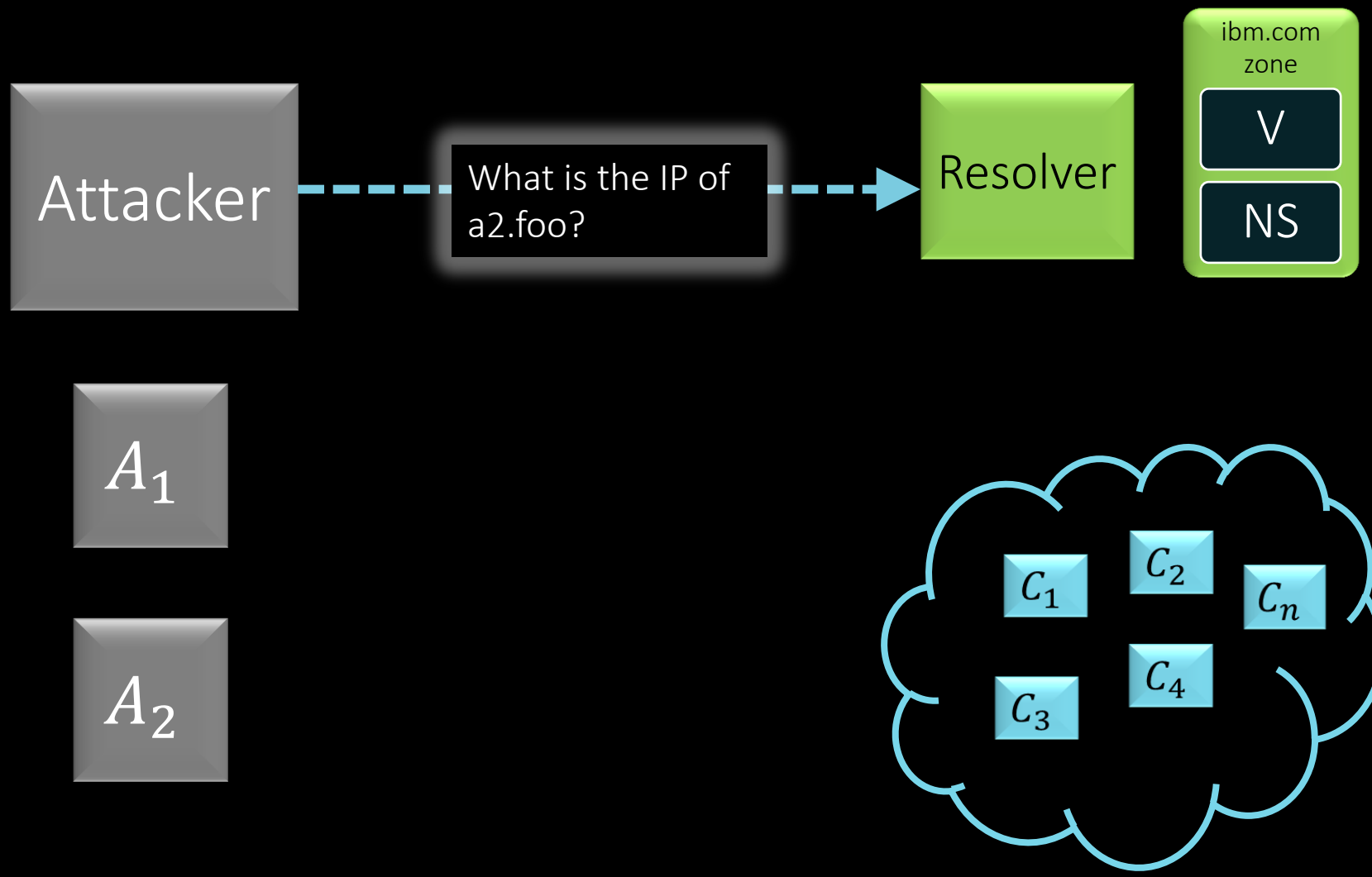
Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	21	[I]

Next list (SRTT sorted)

A_2

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

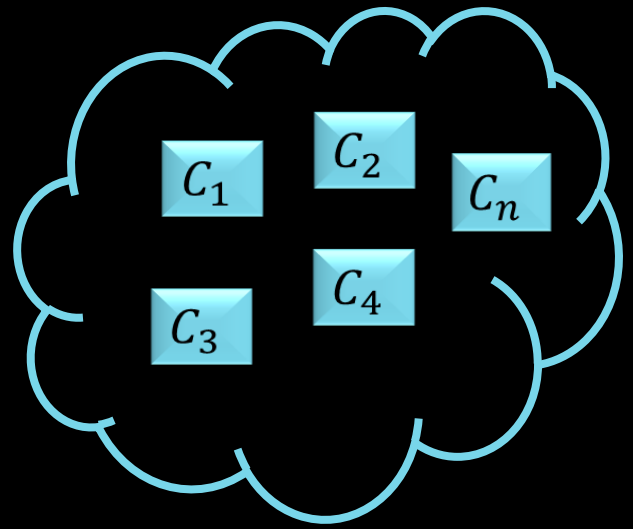
A_2

Resolver

ibm.com zone

V

NS



Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	21	[I]

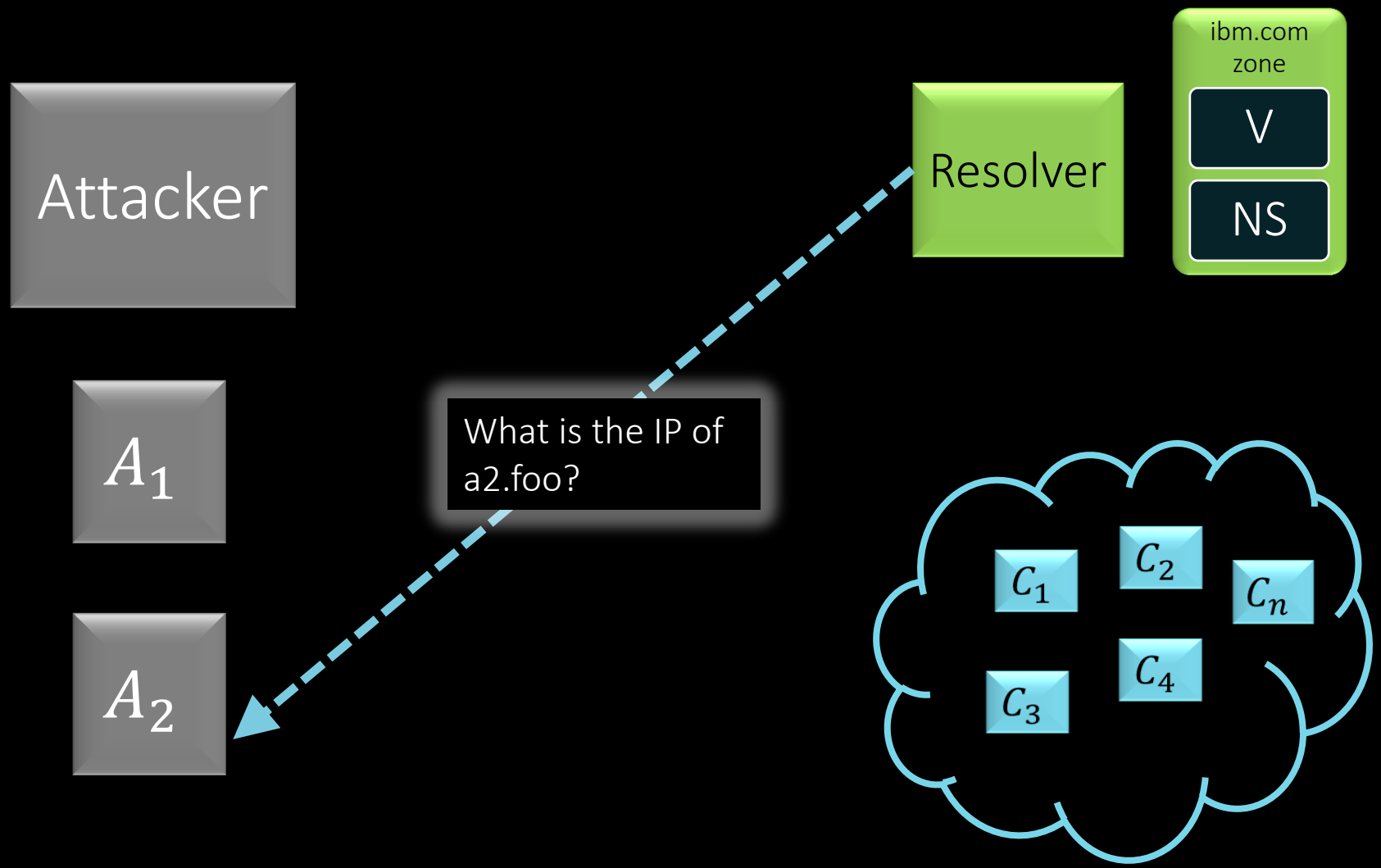
Next list (SRTT sorted)

A_2

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	21	[I]

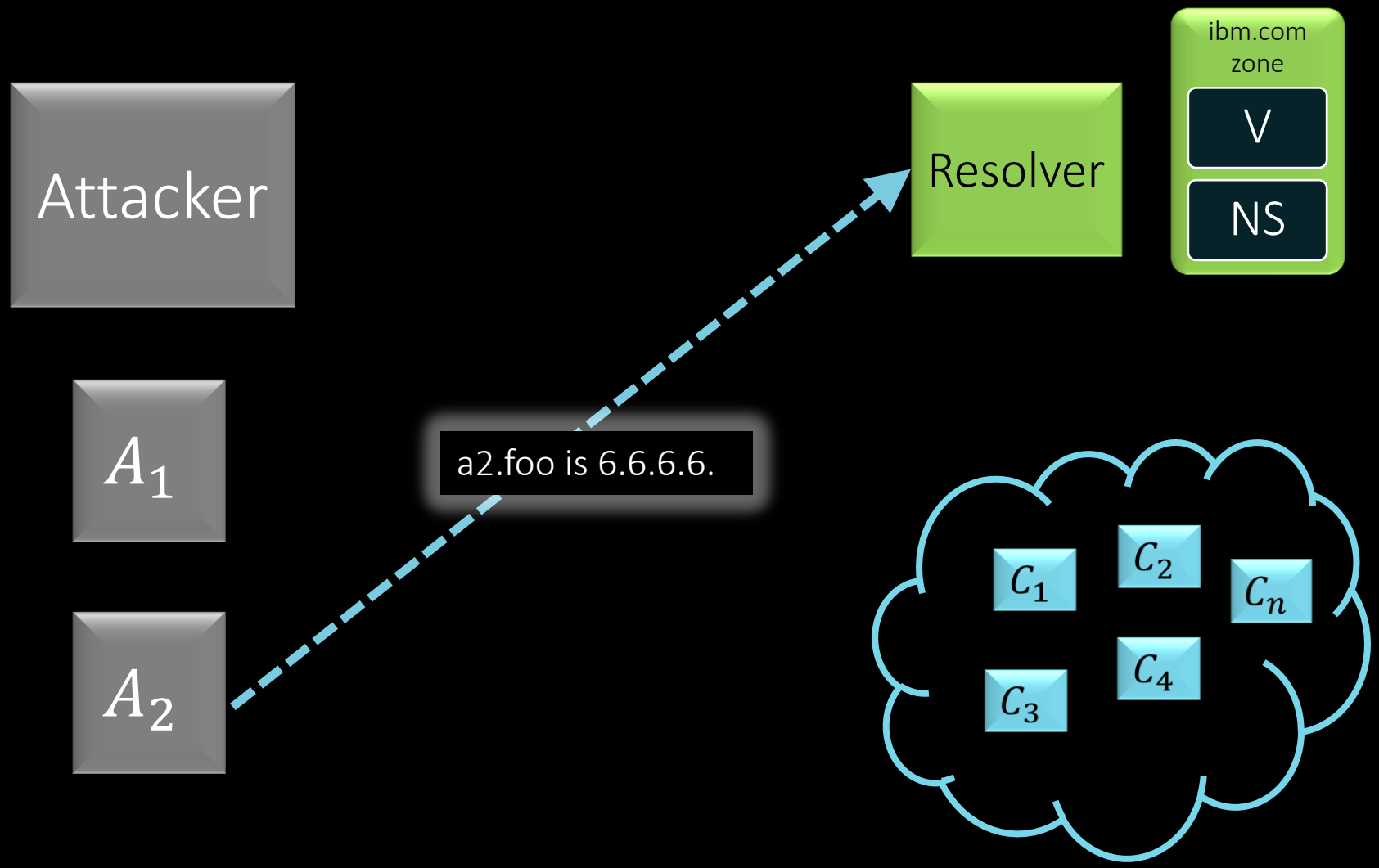
Next list (SRTT sorted)

A_2

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

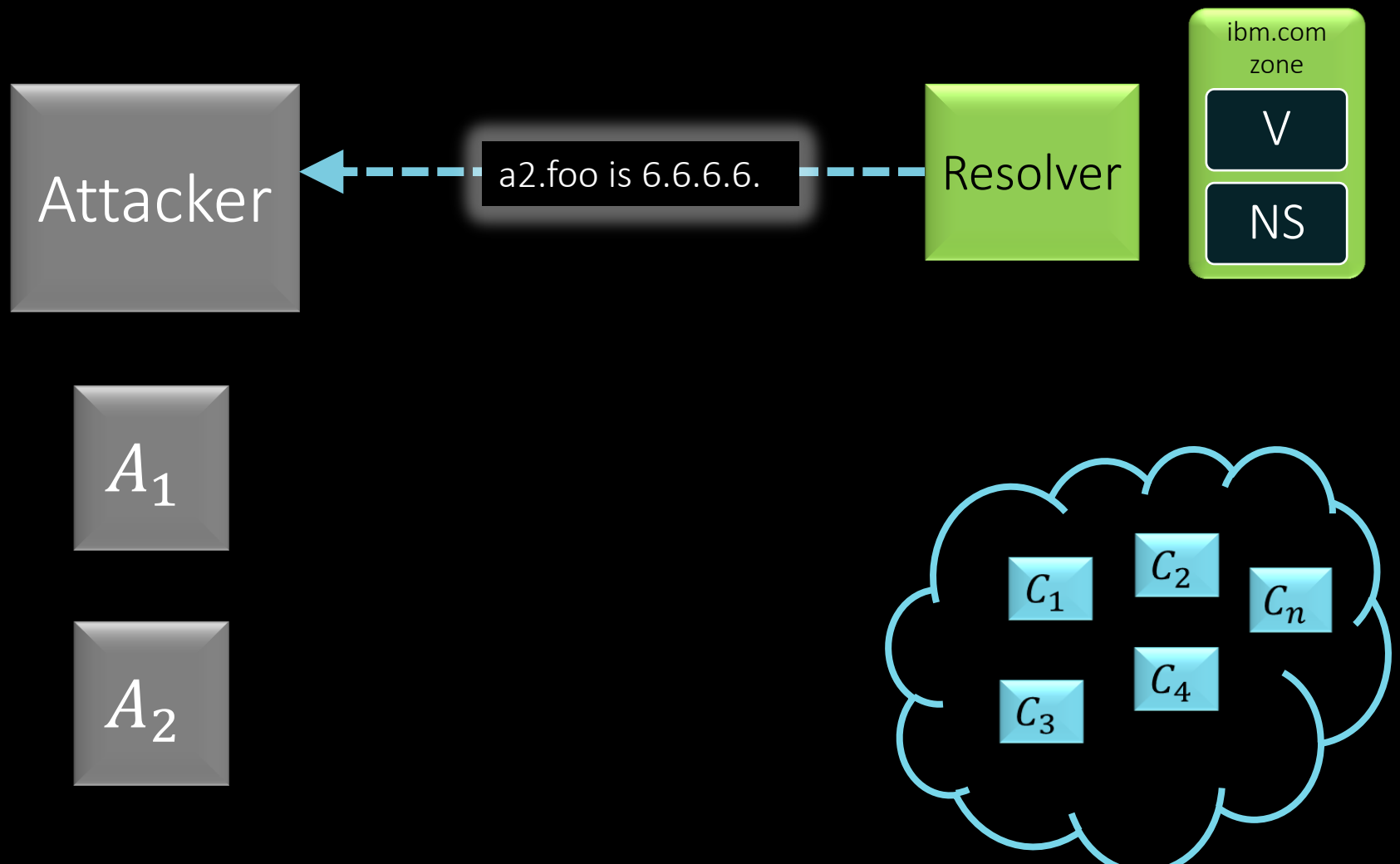
WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	10000	[U]

Next list (SRTT sorted)

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

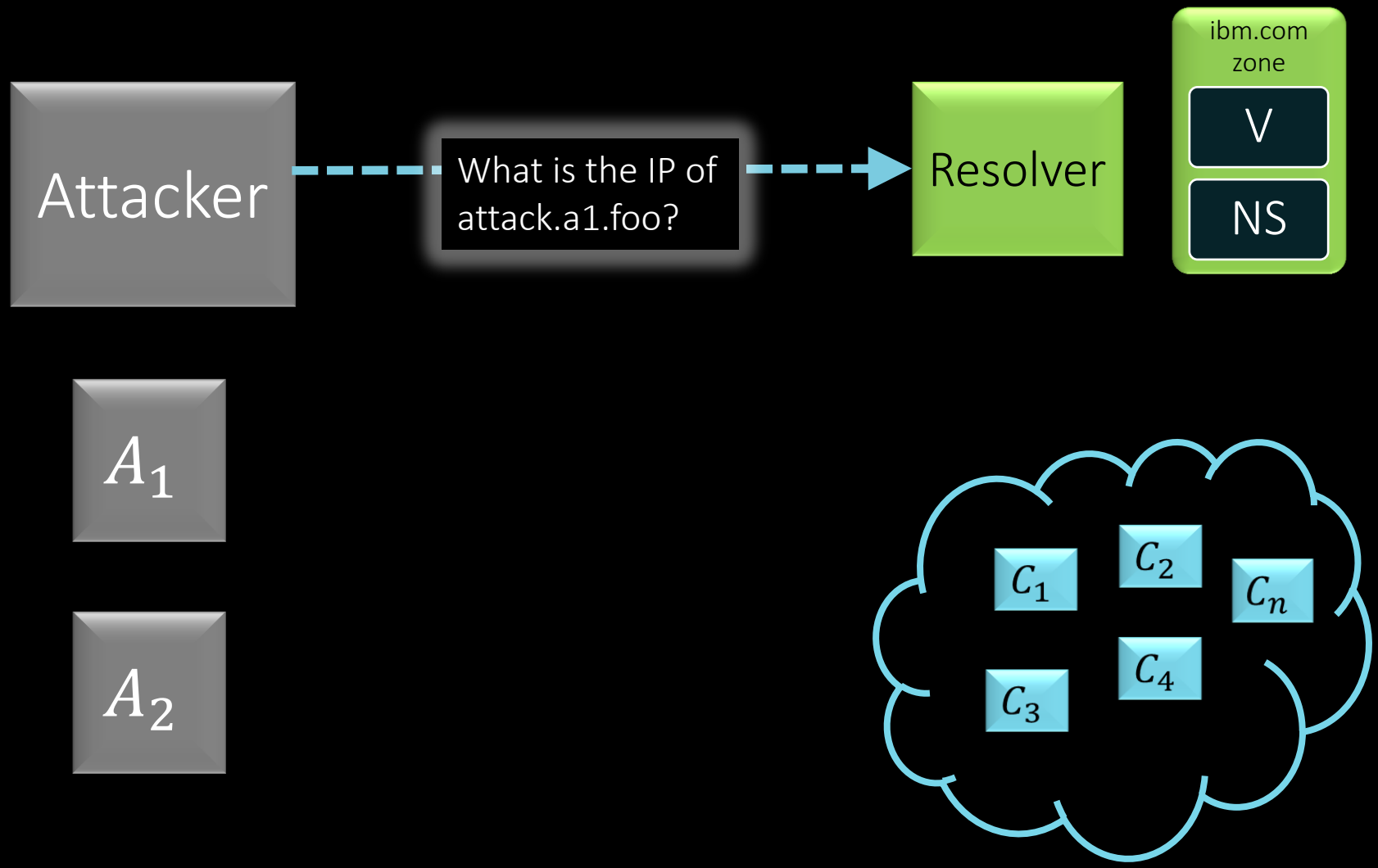
WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	10000	[U]

Next list (SRTT sorted)

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

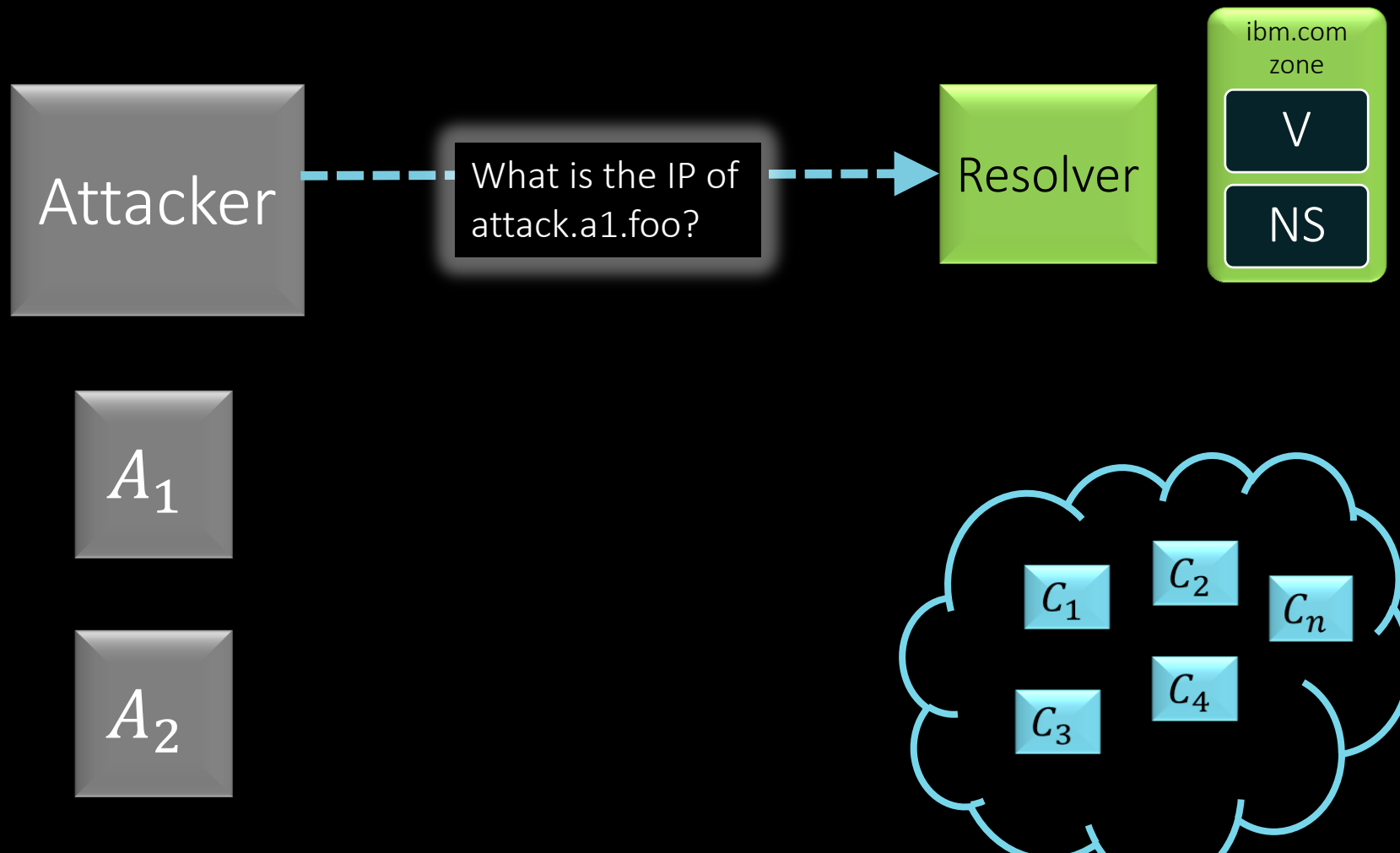
WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	10000	[U]

Next list (SRTT sorted)

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	10000	[U]
A_1	31	[I]

Next list (SRTT sorted)

A_1

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

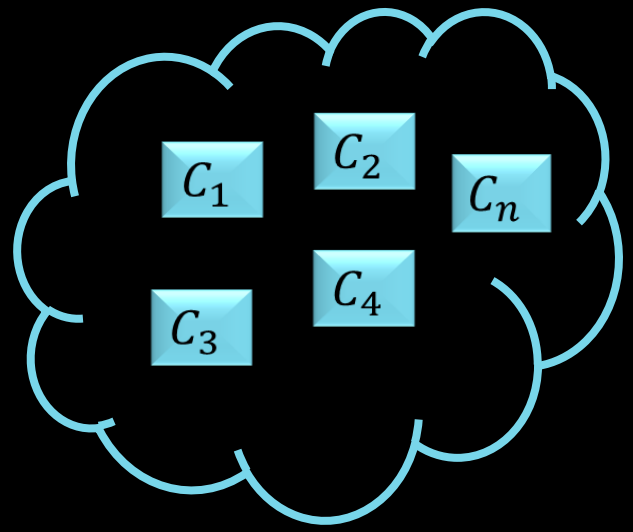
A_2

Resolver

ibm.com zone

V

NS



Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	10000	[U]
A_1	31	[I]

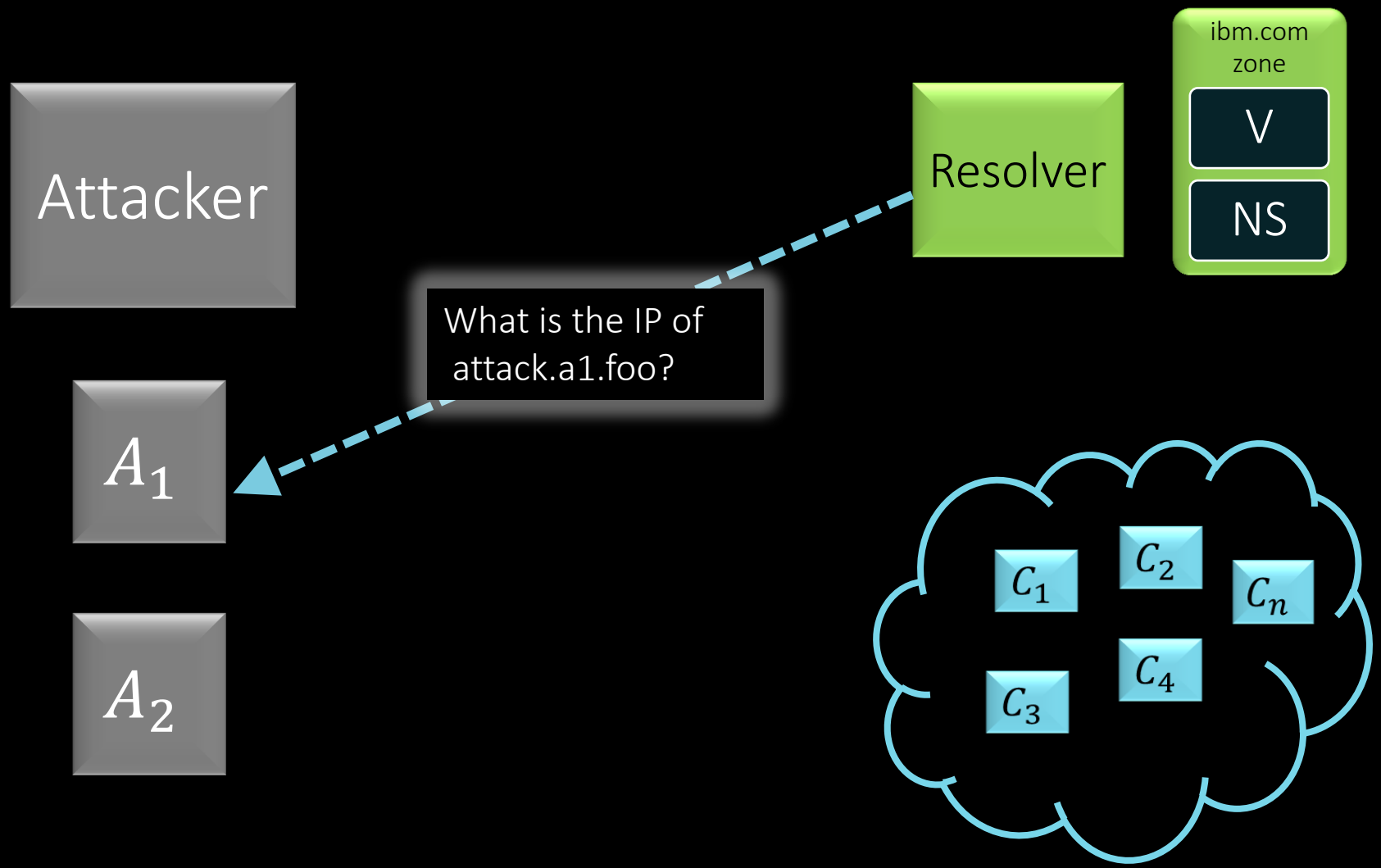
Next list (SRTT sorted)

A_1

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	10000	[U]
A_1	31	[I]

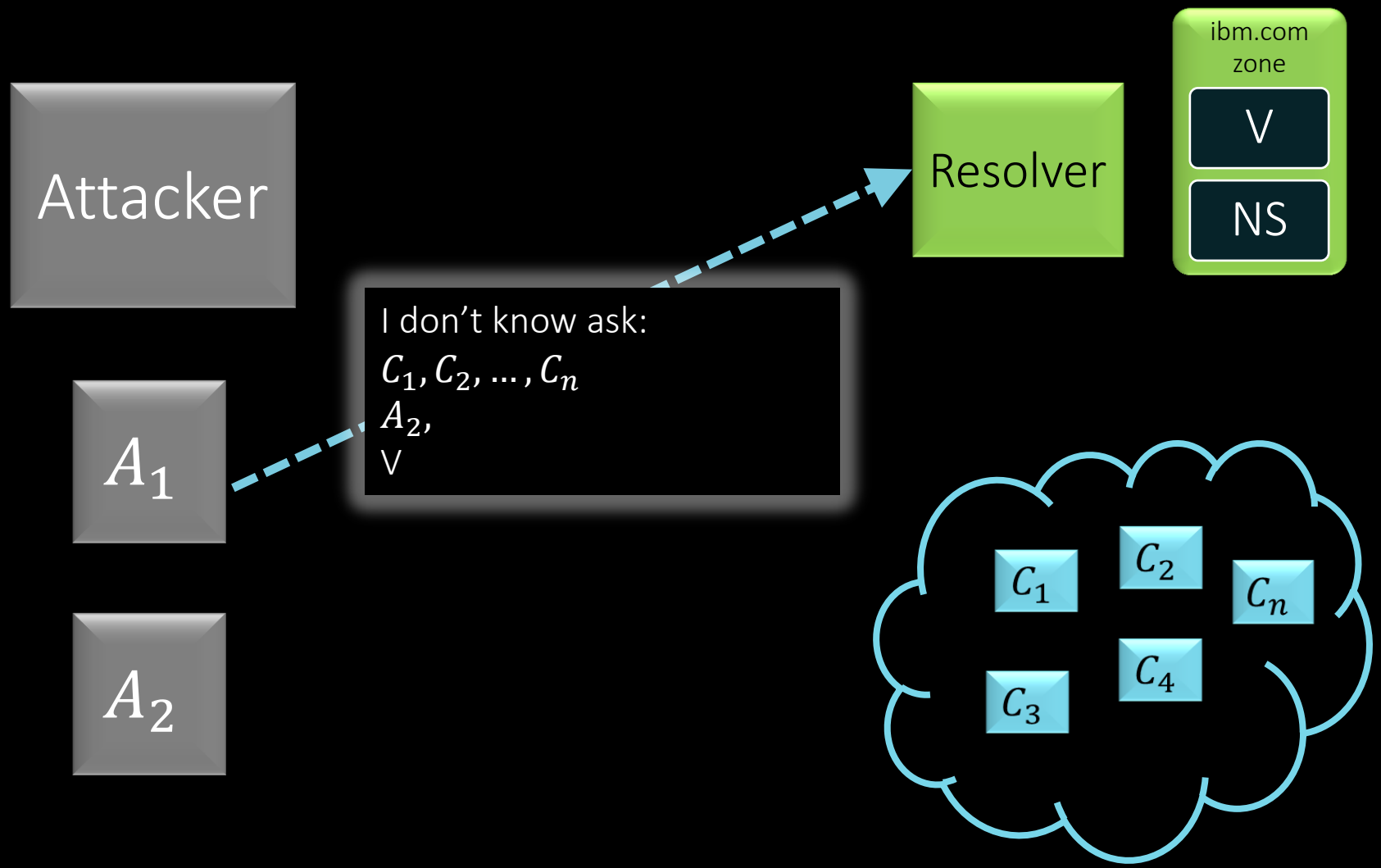
Next list (SRTT sorted)

A_1

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	10000	[U]
A_1	78443	[U]

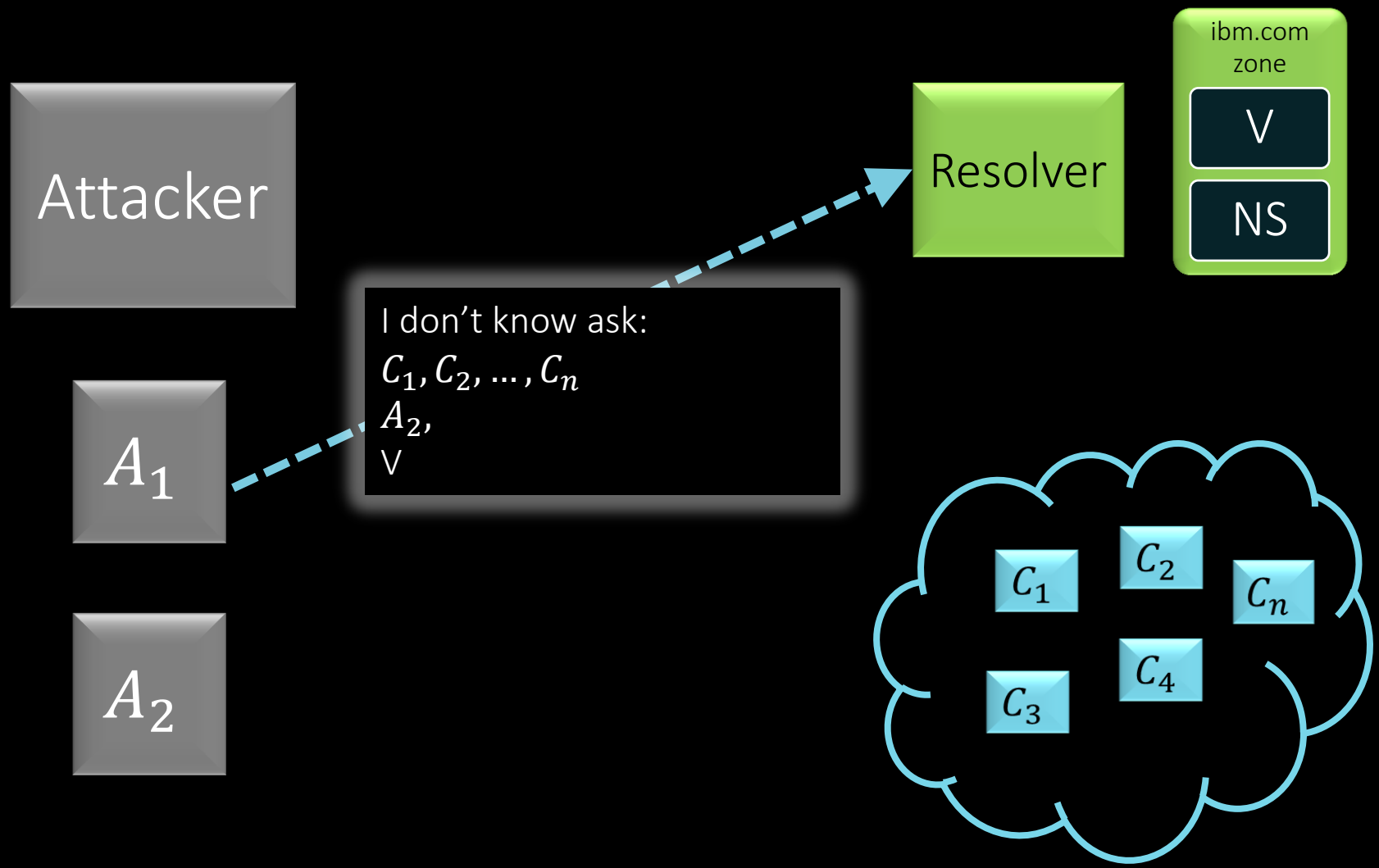
Next list (SRTT sorted)

A_1

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	10000	[U]
A_1	78443	[U]
C_1	23	[I]
C_2	22	[I]
⋮		
C_k	32	[I]
⋮		
C_n	25	[I]

Next list (SRTT sorted)

C_2 → C_1 → … → C_k → A_2 → V

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

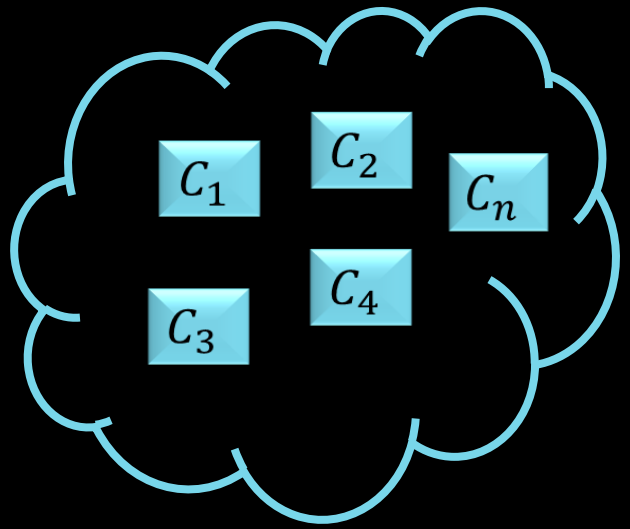
A_2

Resolver

ibm.com zone

V

NS



Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	10000	[U]
A_1	78443	[U]
C_1	23	[I]
C_2	22	[I]
⋮		
C_k	32	[I]
⋮		
C_n	25	[I]

Next list (SRTT sorted)

C_2 → C_1 → … → C_k → A_2 → V

SRTT Operations

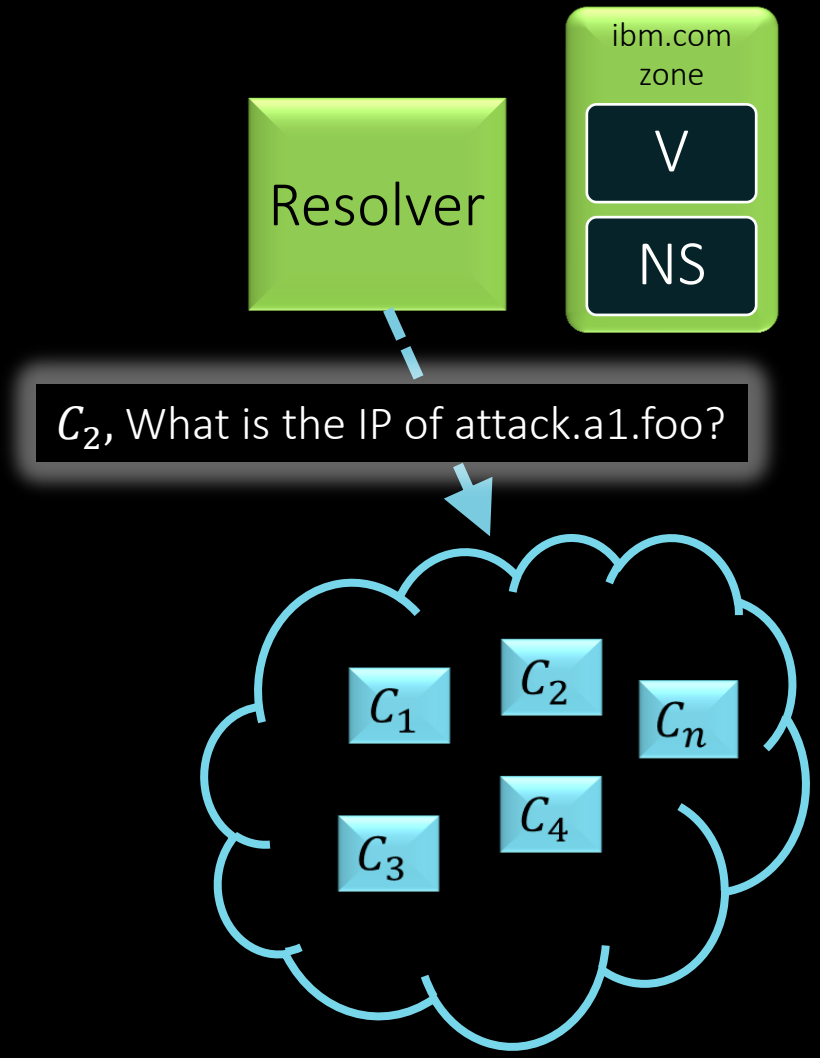
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	100000	[U]
NS	90000	[U]
A_2	10000	[U]
A_1	78443	[U]
C_1	23	[I]
C_2	22	[I]
⋮		
C_k	32	[I]
⋮		
C_n	25	[I]

Next list (SRTT sorted)

C_2 → C_1 → … → C_k → A_2 → V

SRTT Operations

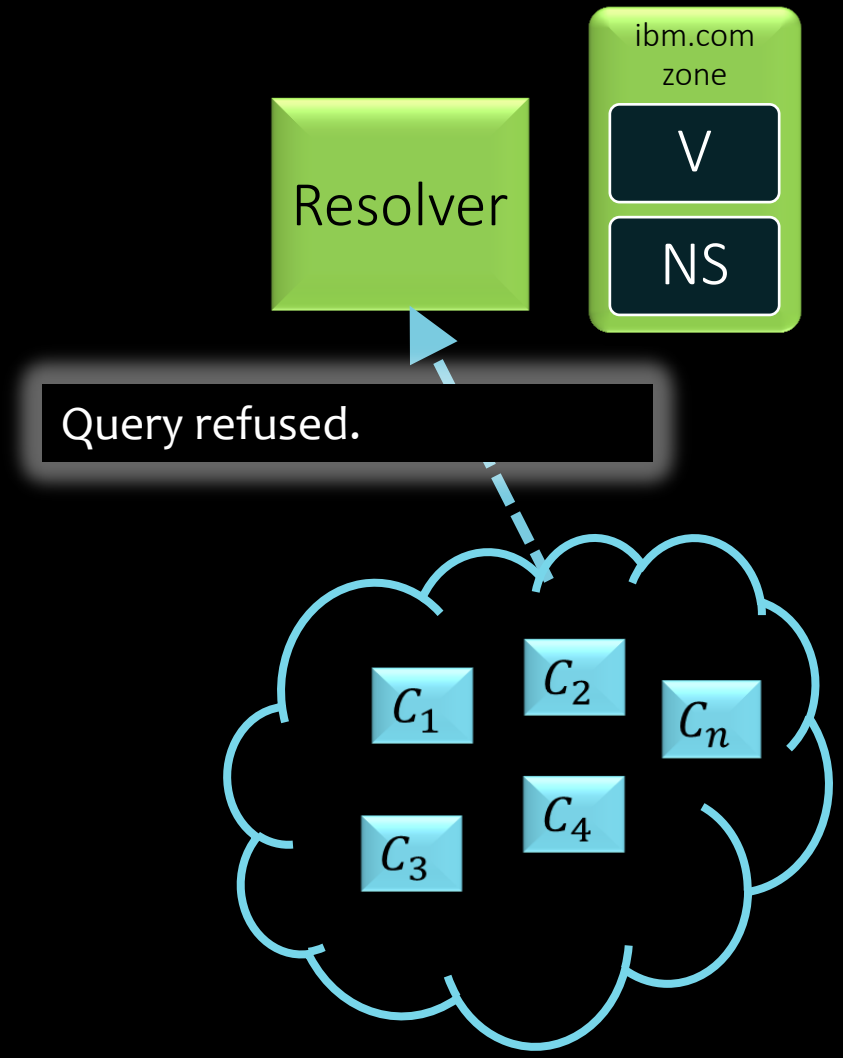
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	98000	[D]
NS	90000	[U]
A_2	9800	[D]
A_1	78443	[U]
C_1	22	[D]
C_2	84341	[U]
⋮		
C_k	31	[D]
⋮		
C_n	24	[D]

Next list (SRTT sorted)

C_2 → C_1 → … → C_k → A_2 → V

SRTT Operations

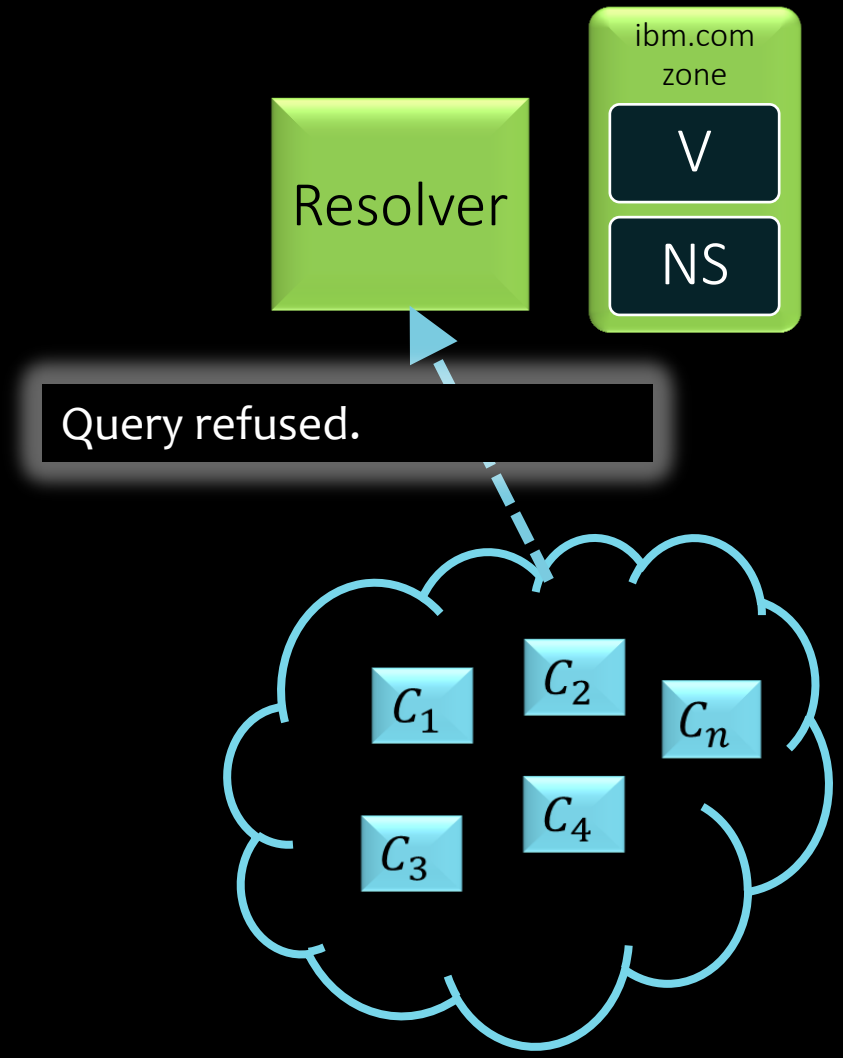
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	98000	[D]
NS	90000	[U]
A_2	9800	[D]
A_1	78443	[U]
C_1	22	[D]
C_2	84341	[U]
⋮		
C_k	31	[D]
⋮		
C_n	24	[D]

Next list (SRTT sorted)

C_1 → ... → C_k → A_2 → V

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

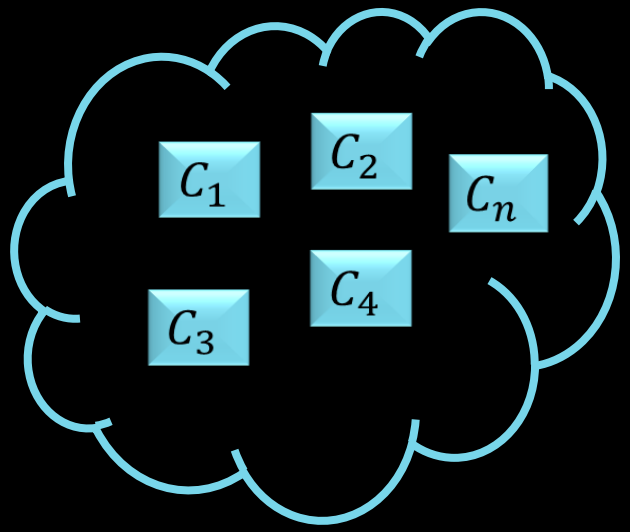
A_2

Resolver

ibm.com zone

V

NS



Resolver's SRTT Cache

WHO	SRTT	OP
V	98000	[D]
NS	90000	[U]
A_2	9800	[D]
A_1	78443	[U]
C_1	22	[D]
C_2	84341	[U]
⋮		
C_k	31	[D]
⋮		
C_n	24	[D]

Next list (SRTT sorted)

C_1 → ... → C_k → A_2 → V

SRTT Operations

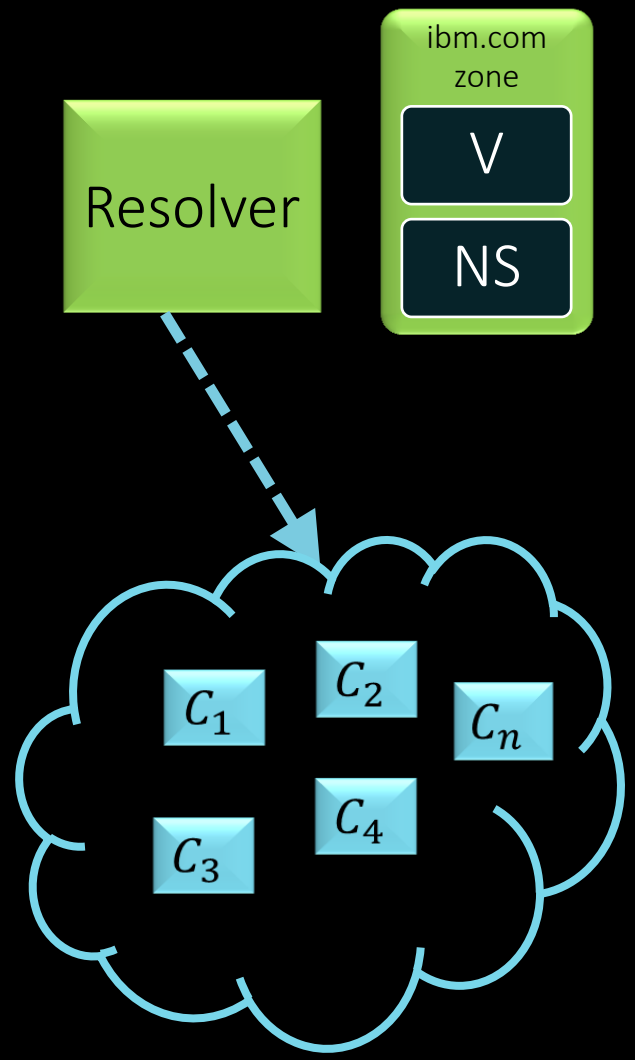
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	98000	[D]
NS	90000	[U]
A_2	9800	[D]
A_1	78443	[U]
C_1	22	[D]
C_2	84341	[U]
⋮		
C_k	31	[D]
⋮		
C_n	24	[D]

Next list (SRTT sorted)

C_1 → ... → C_k → A_2 → V

SRTT Operations

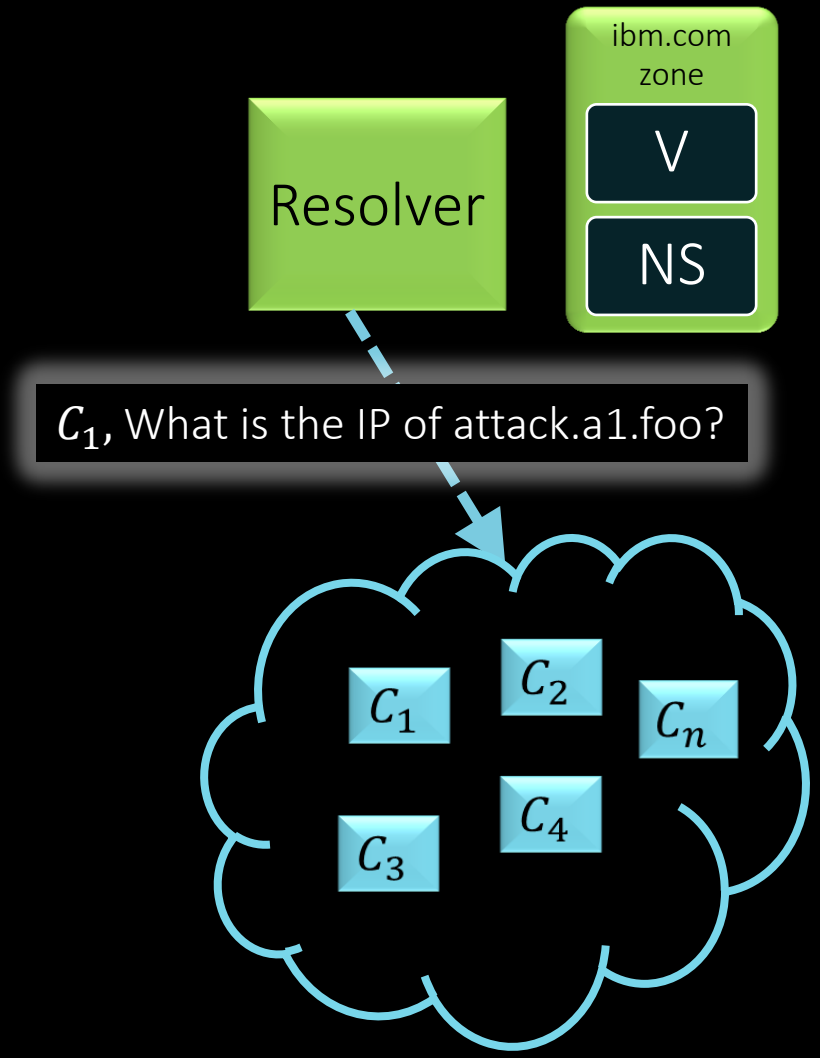
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	98000	[D]
NS	90000	[U]
A_2	9800	[D]
A_1	78443	[U]
C_1	22	[D]
C_2	84341	[U]
⋮		
C_k	31	[D]
⋮		
C_n	24	[D]

Next list (SRTT sorted)

C_1 → ... → C_k → A_2 → V

SRTT Operations

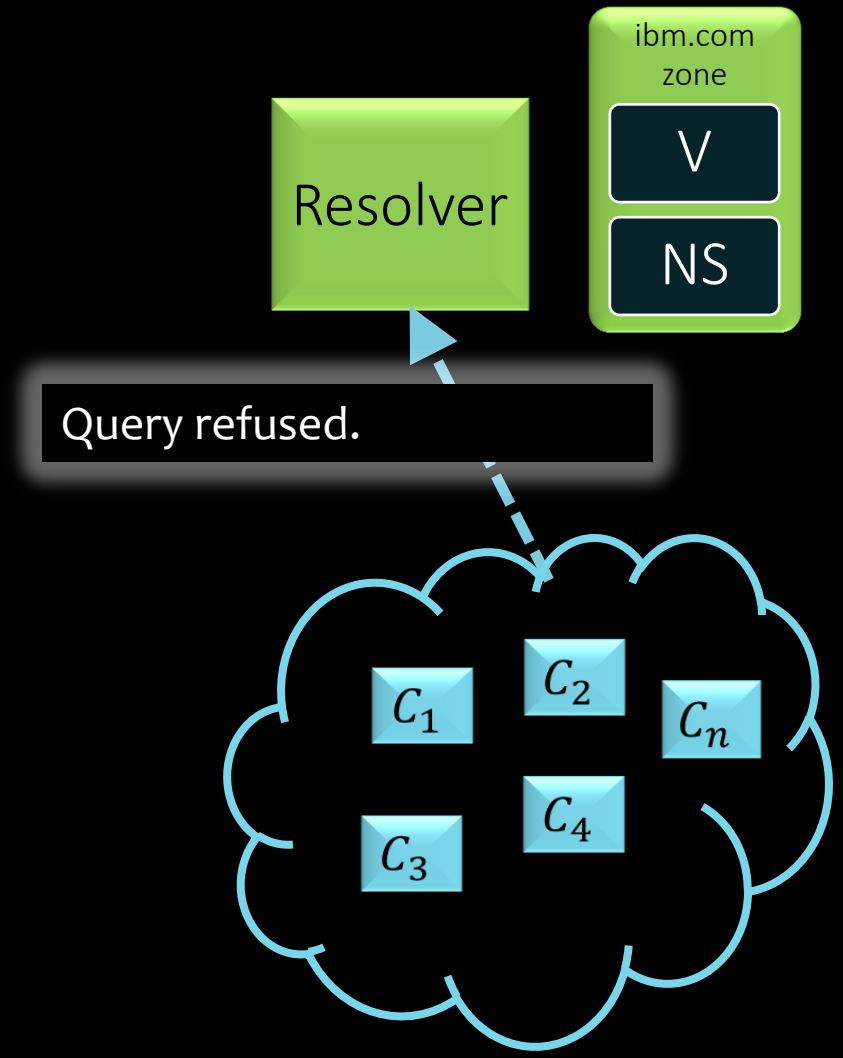
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	96040	[D]
NS	90000	[U]
A_2	9604	[D]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	30	[D]
⋮		
C_n	23	[D]

Next list (SRTT sorted)

C_1 → ... → C_k → A_2 → V

SRTT Operations

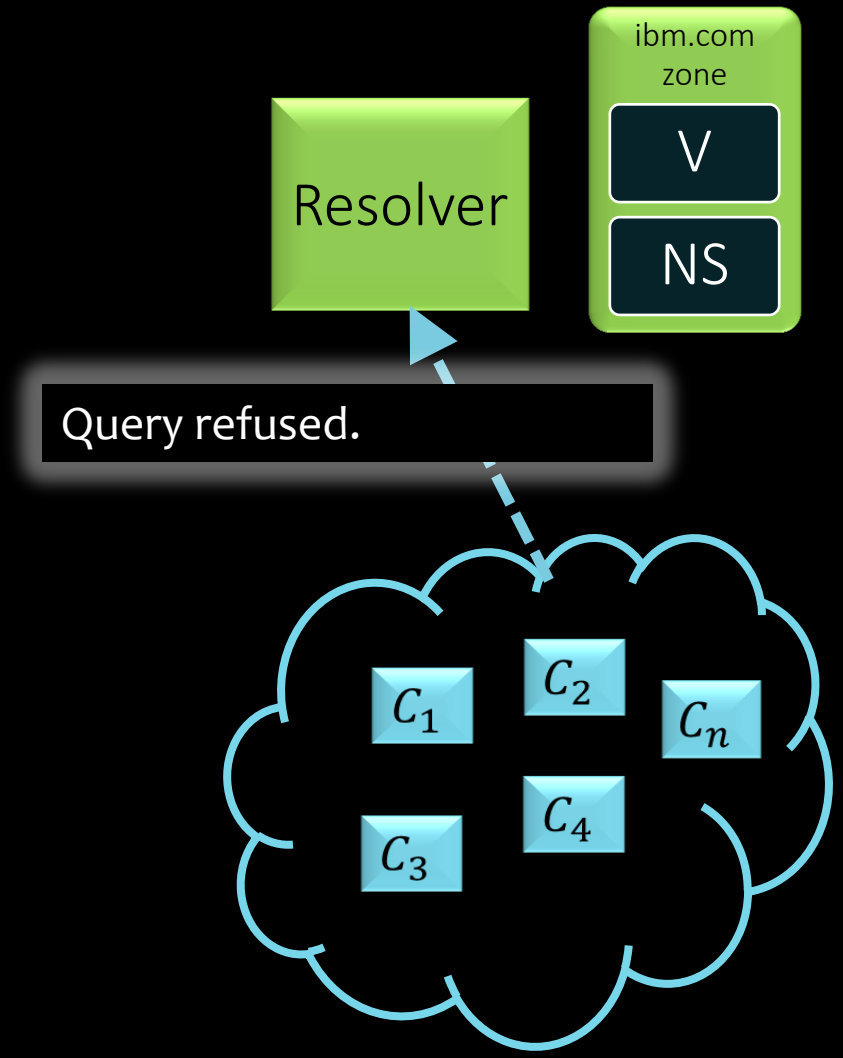
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	96040	[D]
NS	90000	[U]
A_2	9604	[D]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	30	[D]
⋮		
C_n	23	[D]

Next list (SRTT sorted)

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

After $n - 1$ Iterations…

The Attack

Attacker

A_1

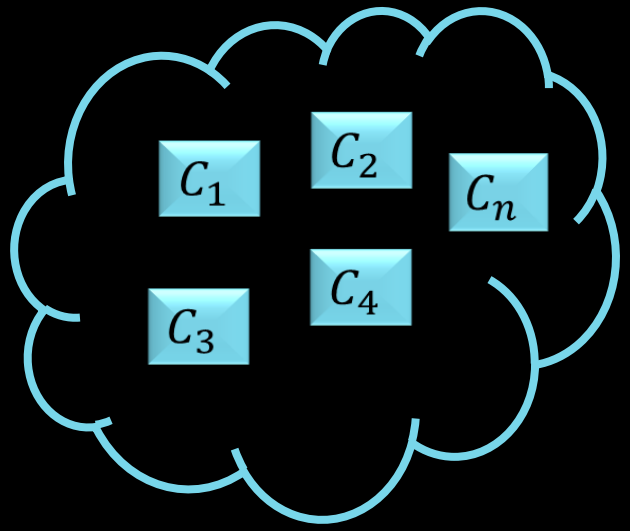
A_2

Resolver

ibm.com zone

V

NS



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^{n-1}$	[D]
NS	90000	[U]
A_2	$10000 \cdot 0.98^{n-1}$	[D]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	$32 \cdot 0.98^{n-1}$	[D]
⋮		
C_n	53248	[U]

Next list (SRTT sorted)

C_k → A_2 → V

SRTT Operations

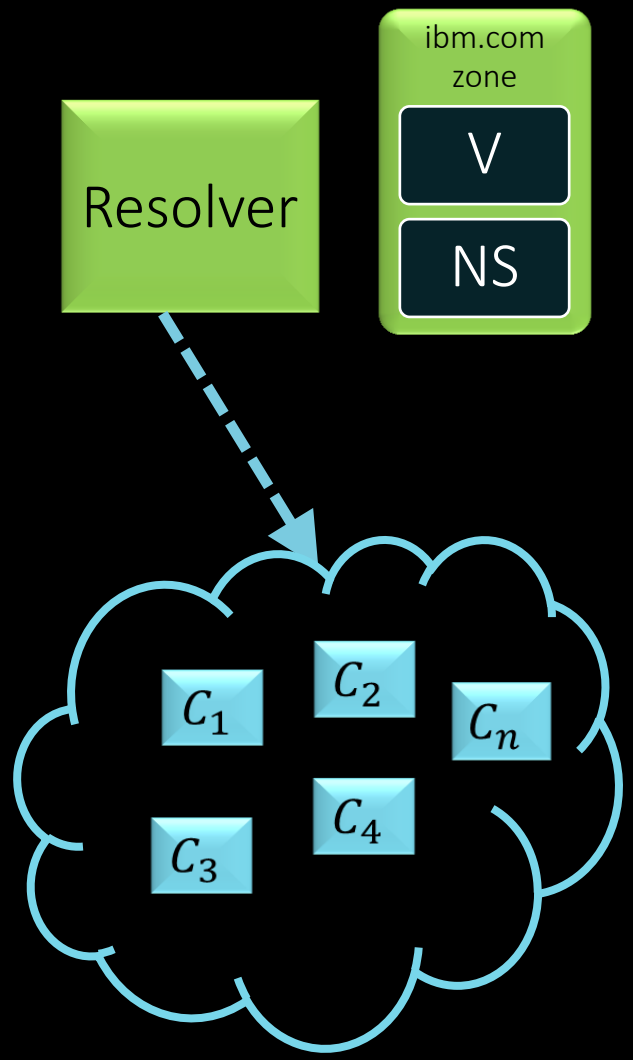
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^{n-1}$	[D]
NS	90000	[U]
A_2	$10000 \cdot 0.98^{n-1}$	[D]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	$32 \cdot 0.98^{n-1}$	[D]
⋮		
C_n	53248	[U]

Next list (SRTT sorted)

C_k → A_2 → V

SRTT Operations

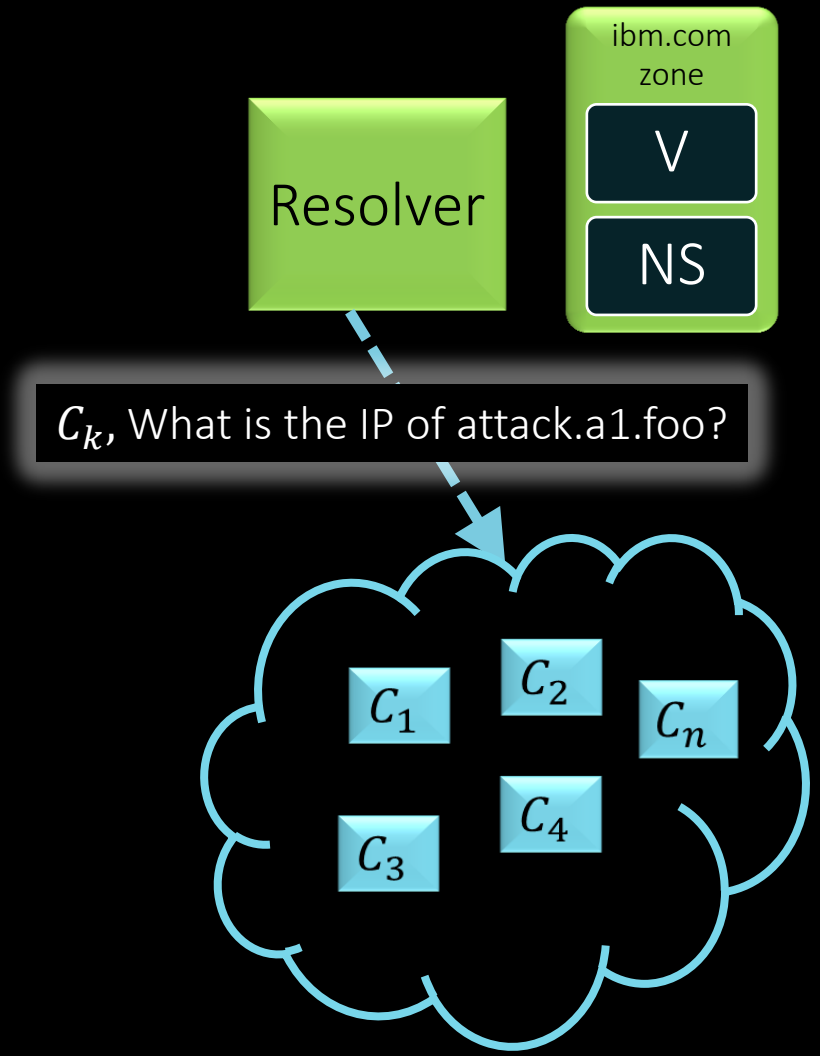
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^{n-1}$	[D]
NS	90000	[U]
A_2	$10000 \cdot 0.98^{n-1}$	[D]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	$32 \cdot 0.98^{n-1}$	[D]
⋮		
C_n	53248	[U]

Next list (SRTT sorted)

C_k → A_2 → V

SRTT Operations

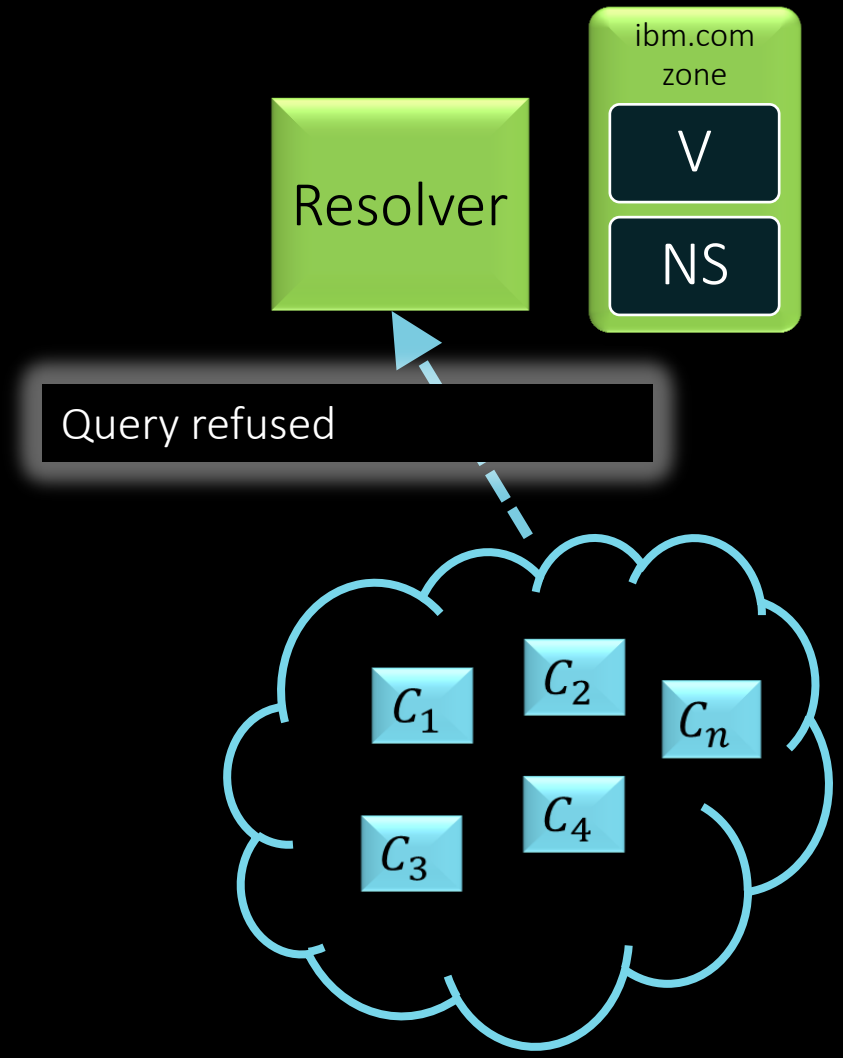
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^n$	[D]
NS	90000	[U]
A_2	$10000 \cdot 0.98^n$	[D]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	91203	[U]
⋮		
C_n	53248	[U]

Next list (SRTT sorted)

C_k → A_2 → V

SRTT Operations

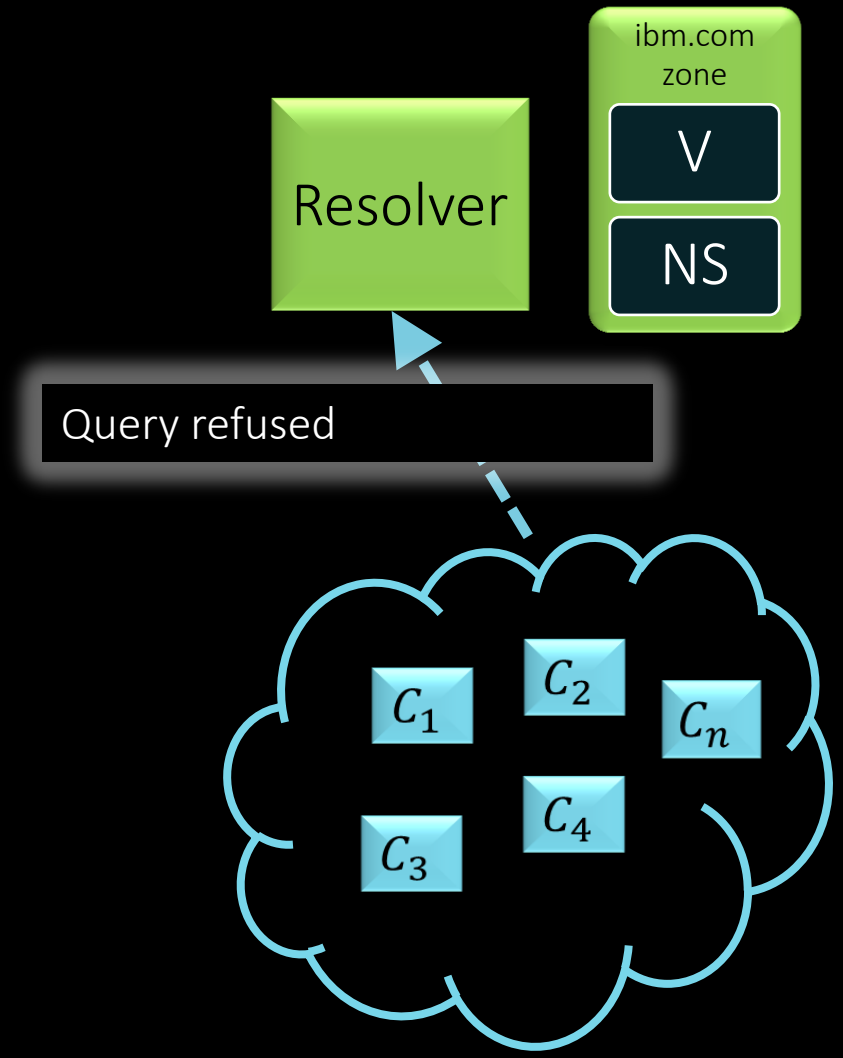
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^n$	[D]
NS	90000	[U]
A_2	$10000 \cdot 0.98^n$	[D]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	91203	[U]
⋮		
C_n	53248	[U]

Next list (SRTT sorted)

C_k → A_2 → V

SRTT Operations

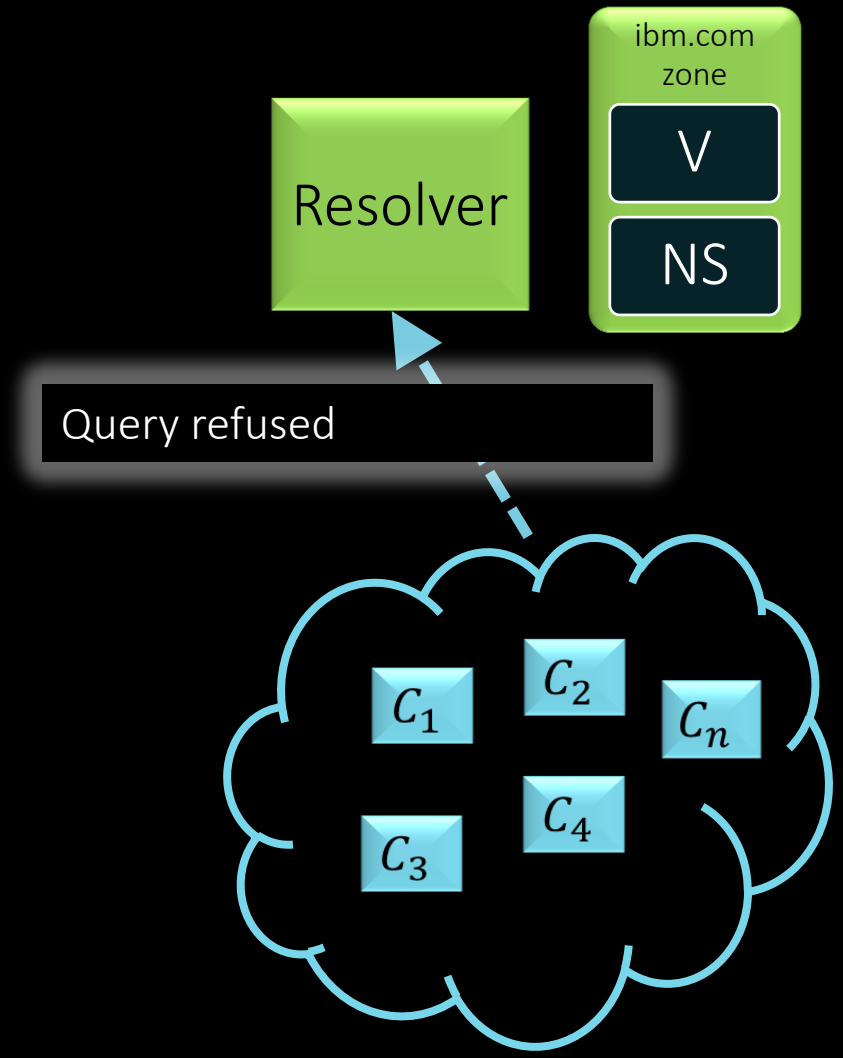
[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

A_2



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^n$	[D]
NS	90000	[U]
A_2	$10000 \cdot 0.98^n$	[D]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	91203	[U]
⋮		
C_n	53248	[U]

Next list (SRTT sorted)

A_2 → V

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

A_1

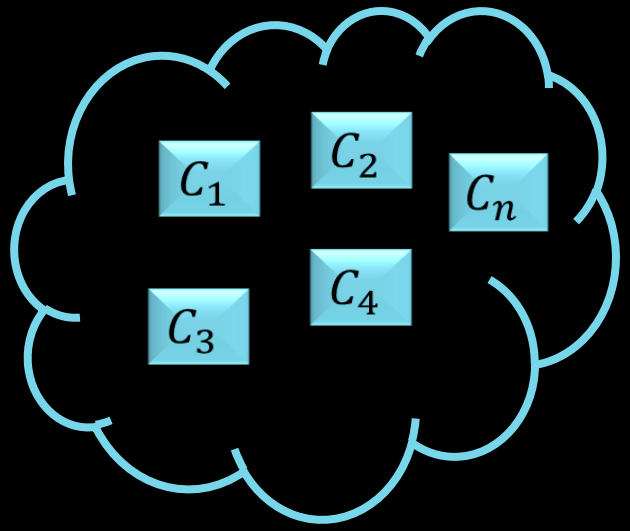
A_2

Resolver

ibm.com zone

V

NS



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^n$	[D]
NS	90000	[U]
A_2	$10000 \cdot 0.98^n$	[D]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	91203	[U]
⋮		
C_n	53248	[U]

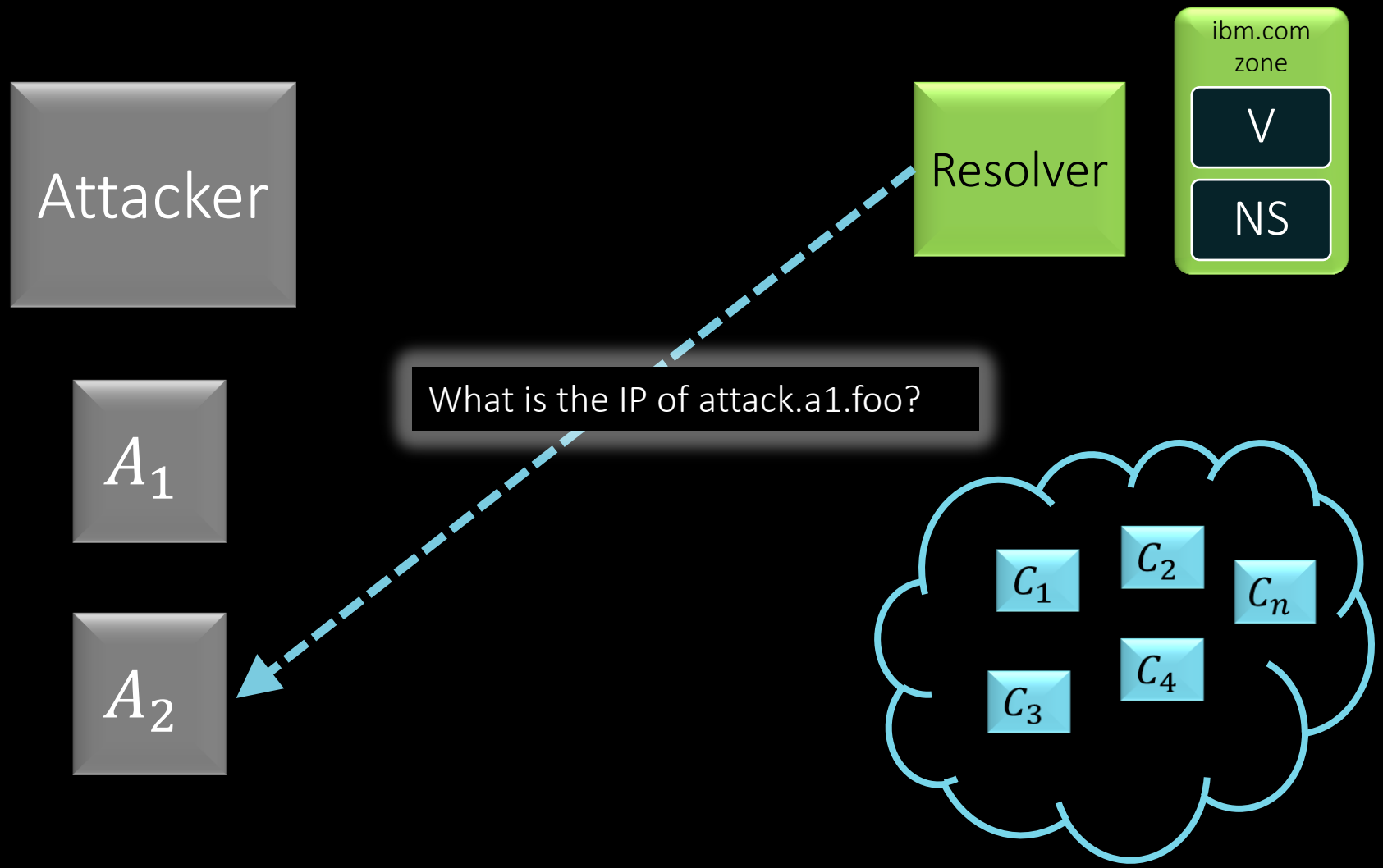
Next list (SRTT sorted)

A_2 → V

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^n$	[D]
NS	90000	[U]
A_2	$10000 \cdot 0.98^n$	[D]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	91203	[U]
⋮		
C_n	53248	[U]

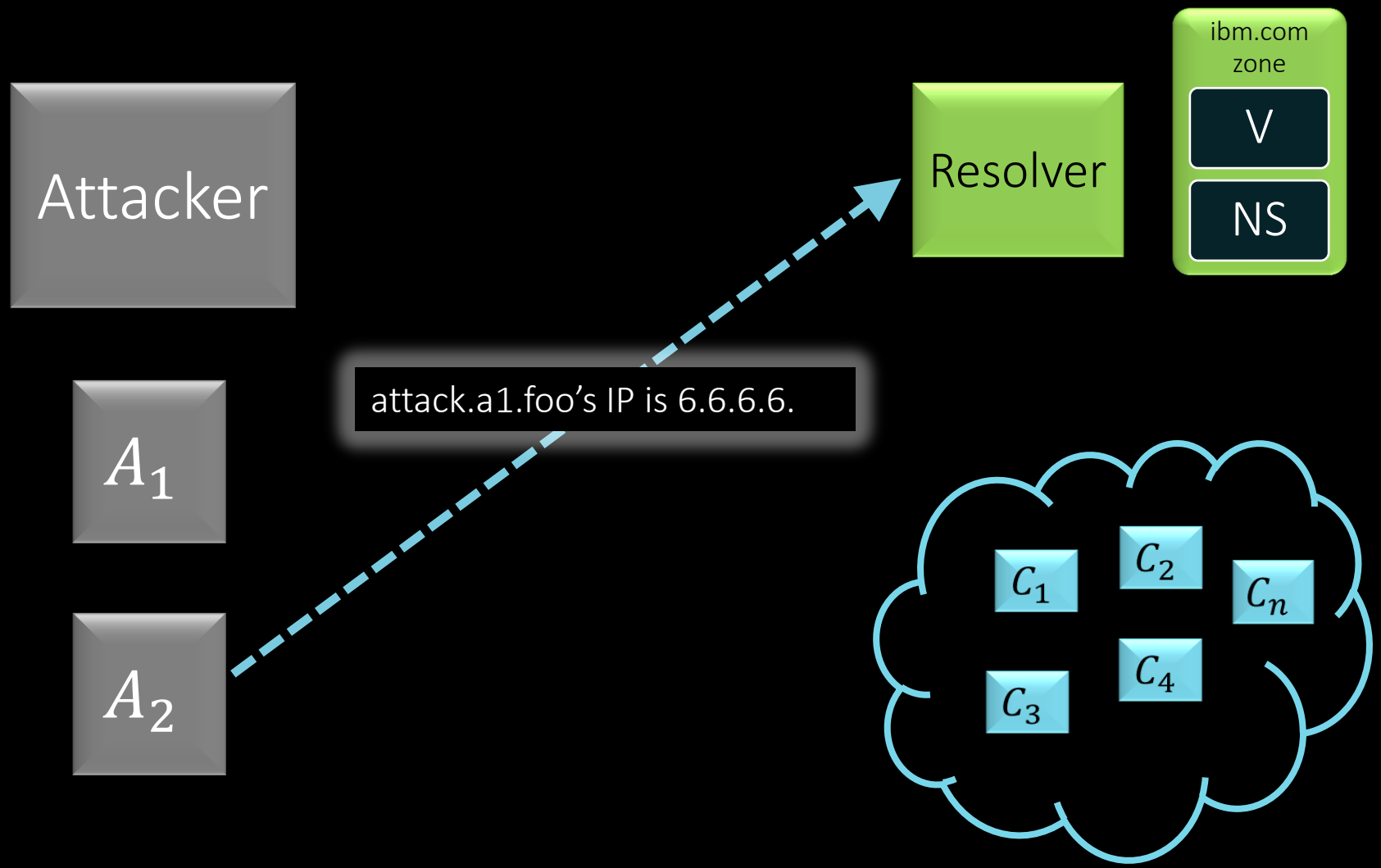
Next list (SRTT sorted)

A_2 → V

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^{n+1}$	[D]
NS	90000	[U]
A_2	12003	[U]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	91203	[U]
⋮		
C_n	53248	[U]

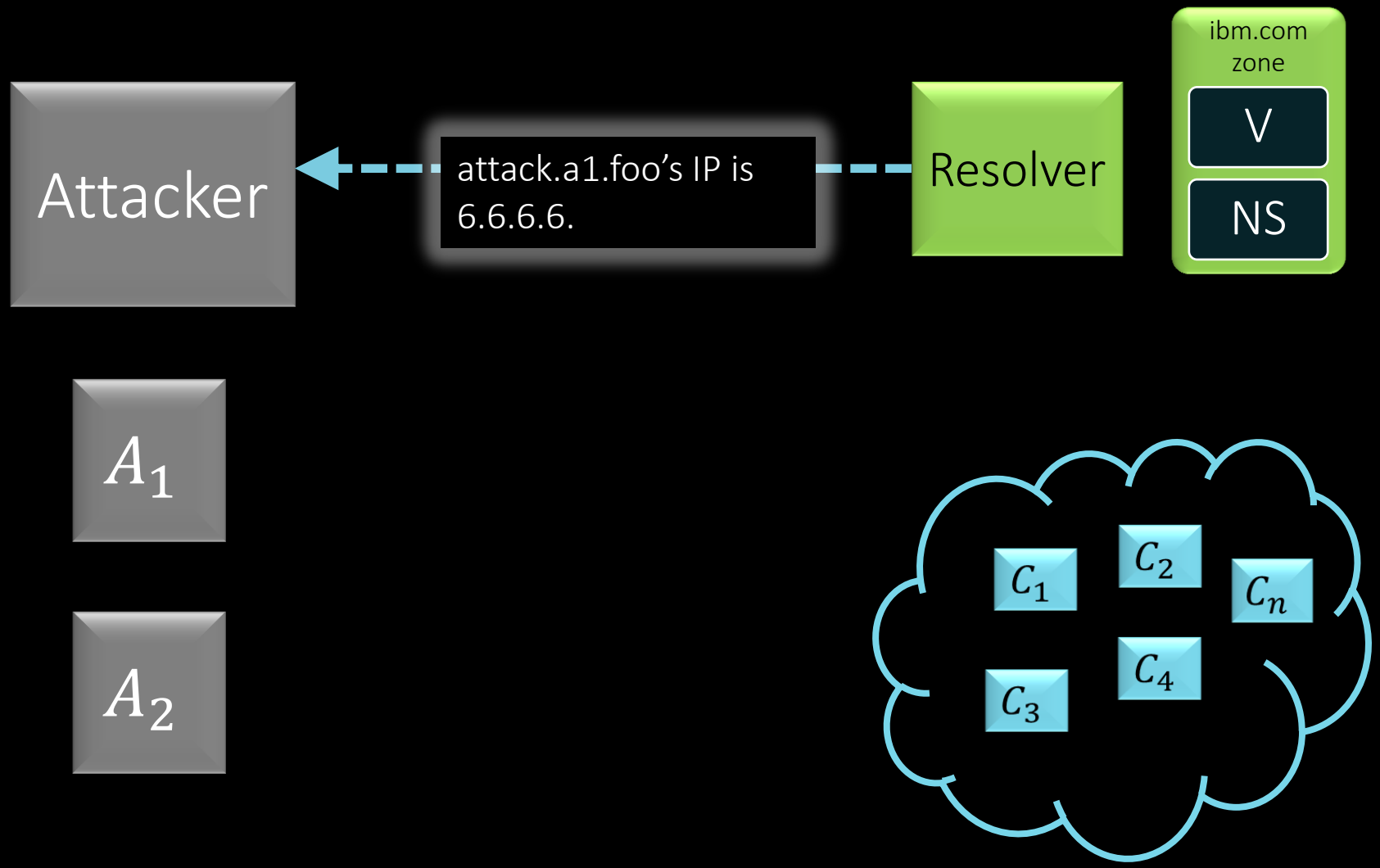
Next list (SRTT sorted)

A_2 → V

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^{n+1}$	[D]
NS	90000	[U]
A_2	12003	[U]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	91203	[U]
⋮		
C_n	53248	[U]

Next list (SRTT sorted)

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

The Attack

Attacker

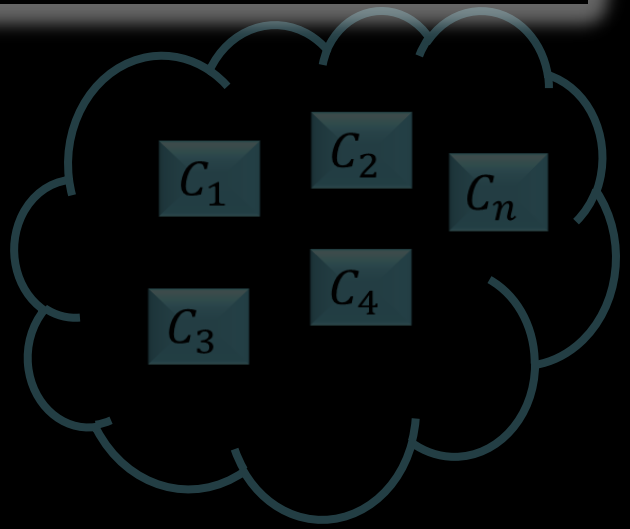
A_1

A_2

ibm.com zone

at 6.

For a sufficiently large value of n, the attacked resolver times out (after 30 seconds), so we don't even need A_2.



Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^{n+1}$	[D]
NS	90000	[U]
A_2	12003	[U]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	91203	[U]
⋮		
C_n	53248	[U]

Next list (SRTT sorted)

SRTT Operations

[I]nit [U]date [D]ecay [E]rror

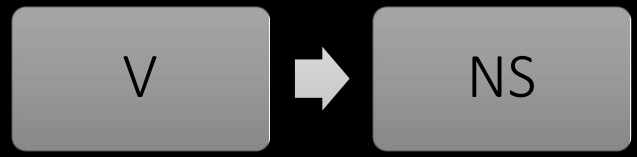
Wrap-up

- We lowered the SRTT value of an arbitrary NS to an arbitrary value.
- Cool features of the attack:
  - The attack is deterministic and requires 3 packets only.
  - We abuse non-open resolvers in contrast to many attacks that abuse open ones.
  - Recovery is not instant as of the SRTT update operation.
- The general lesson is to separate the cache. Never maintain a shared one.

Resolver's SRTT Cache

WHO	SRTT	OP
V	$100000 \cdot 0.98^{n+1}$	[D]
NS	90000	[U]
A_2	12003	[U]
A_1	78443	[U]
C_1	63289	[U]
C_2	84341	[U]
⋮		
C_k	91203	[U]
⋮		
C_n	53248	[U]

Next list (SRTT sorted)



SRTT Operations

[I]nit [U]date [D]ecay [E]rror


?
Thank you.