CSE227 – Graduate Computer Security

DNS

UC San Diego

Housekeeping

General course things to know

- Midpoint check-in document is due **TOMORROW at 11:59pm PT**
  - Introduction (frame the problem)
  - Related work section (should include ~5 – 10 relevant papers)
  - Research plan, current status, what's left to do
  - Submission is on Gradescope from *Canvas*
- Optional meetings with meet available next week on 20th

Today's lecture

Learning Objectives

- Learn what the domain name system is, how it works in practice, and how it really really works in practice
- Discuss the Kaminsky cache poisoning attack, how it works, and why it's still possible today

Preliminaries

What is DNS?

What is DNS?

Domain Name System: Our mechanisms for converting human-readable names to IP addresses.

Why do we have DNS?

Why do we have DNS?

Numbers are hard! Names are easier.

What’s easier to remember? 75.2.44.127, or **ucsd.edu**?

A brief history lesson

DNS back in the day

- There was a single file, called *hosts.txt*, that was run by the Stanford Research Institute for ARPANET membership
- SRI kept the main copy
    - Single place to update records (had to go through someone)
    - People would periodically download hosts.txt, that's how everyone knew what was happen
- <u>What are some problems with this approach?</u>

DNS Intuition

DNS Today

- Rather than centralize everything, we can *decentralize* everything
  - Build a “chain” of knowledge that starts with roots and goes down to the leaves
  - Every step of the way is a “pointer” to the next step, until you get to a final answer
- We can *recursively resolve* names to get to our final answer!
  - And you thought you’d never use recursion…

DNS Hierarchical Namespace

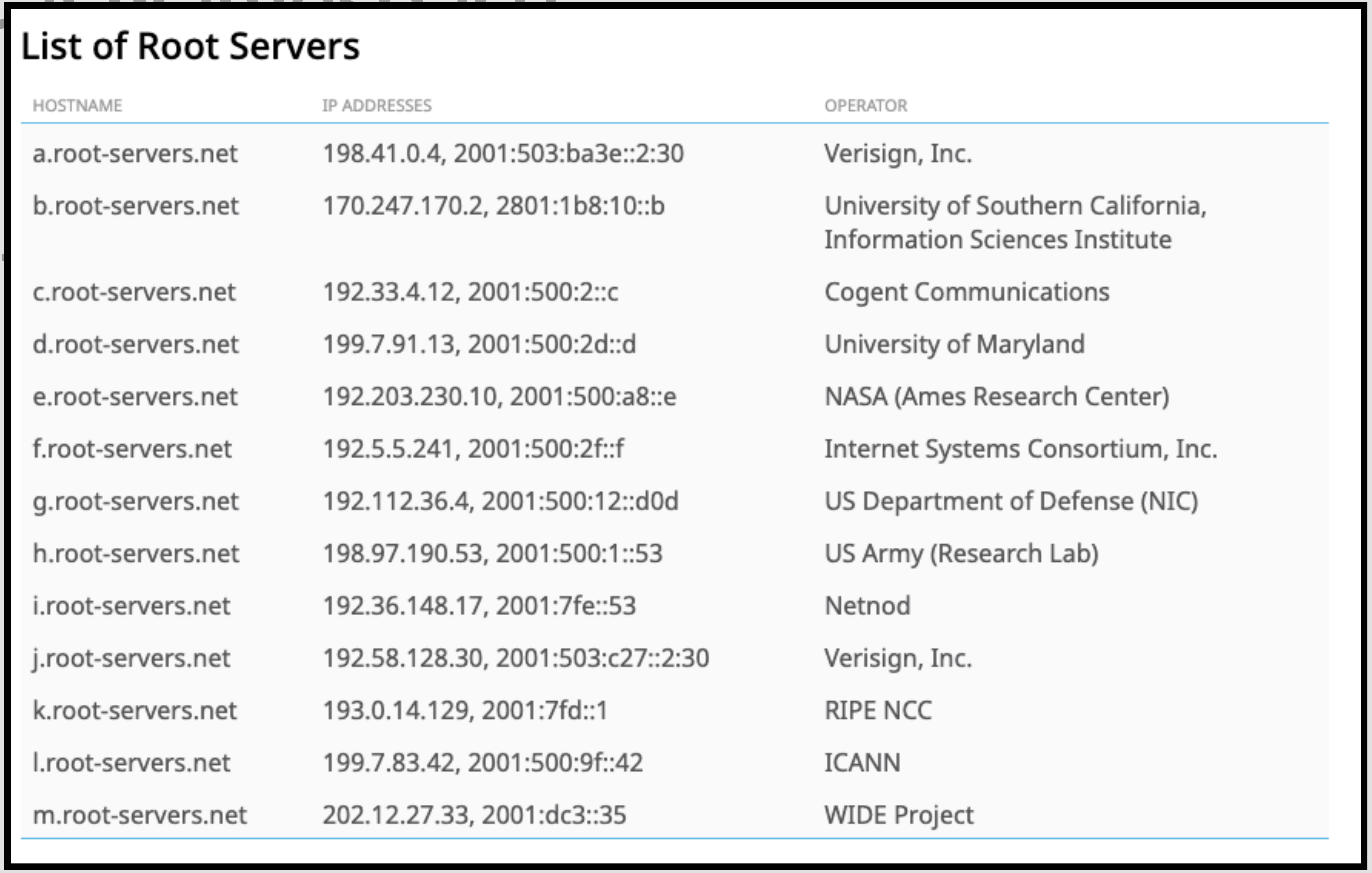
DNS Root *13 root servers*

ICANN

iana
Internet Assigned Numbers Authority

DNS Hierarchical Namespace

DNS Root



List of Root Servers

HOSTNAME	IP ADDRESSES	OPERATOR
a.root-servers.net	198.41.0.4, 2001:503:ba3e::2:30	Verisign, Inc.
b.root-servers.net	170.247.170.2, 2801:1b8:10::b	University of Southern California, Information Sciences Institute
c.root-servers.net	192.33.4.12, 2001:500:2::c	Cogent Communications
d.root-servers.net	199.7.91.13, 2001:500:2d::d	University of Maryland
e.root-servers.net	192.203.230.10, 2001:500:a8::e	NASA (Ames Research Center)
f.root-servers.net	192.5.5.241, 2001:500:2f::f	Internet Systems Consortium, Inc.
g.root-servers.net	192.112.36.4, 2001:500:12::d0d	US Department of Defense (NIC)
h.root-servers.net	198.97.190.53, 2001:500:1::53	US Army (Research Lab)
i.root-servers.net	192.36.148.17, 2001:7fe::53	Netnod
j.root-servers.net	192.58.128.30, 2001:503:c27::2:30	Verisign, Inc.
k.root-servers.net	193.0.14.129, 2001:7fd::1	RIPE NCC
l.root-servers.net	199.7.83.42, 2001:500:9f::42	ICANN
m.root-servers.net	202.12.27.33, 2001:dc3::35	WIDE Project

DNS Hierarchical Namespace

DNS Root	*13 root servers*	ICANN	iana Internet Assigned Numbers Authority
TLD	**.edu, .com,** etc.,	EDUCAUSE	VERISIGN™

DNS Hierarchical Namespace

DNS Root	*13 root servers*	ICANN; iana Internet Assigned Numbers Authority
TLD	**.edu, .com,** etc.,	EDUCAUSE; VERISIGN™
Authoritative	ucsd.edu	UC San Diego

DNS Hierarchical Namespace

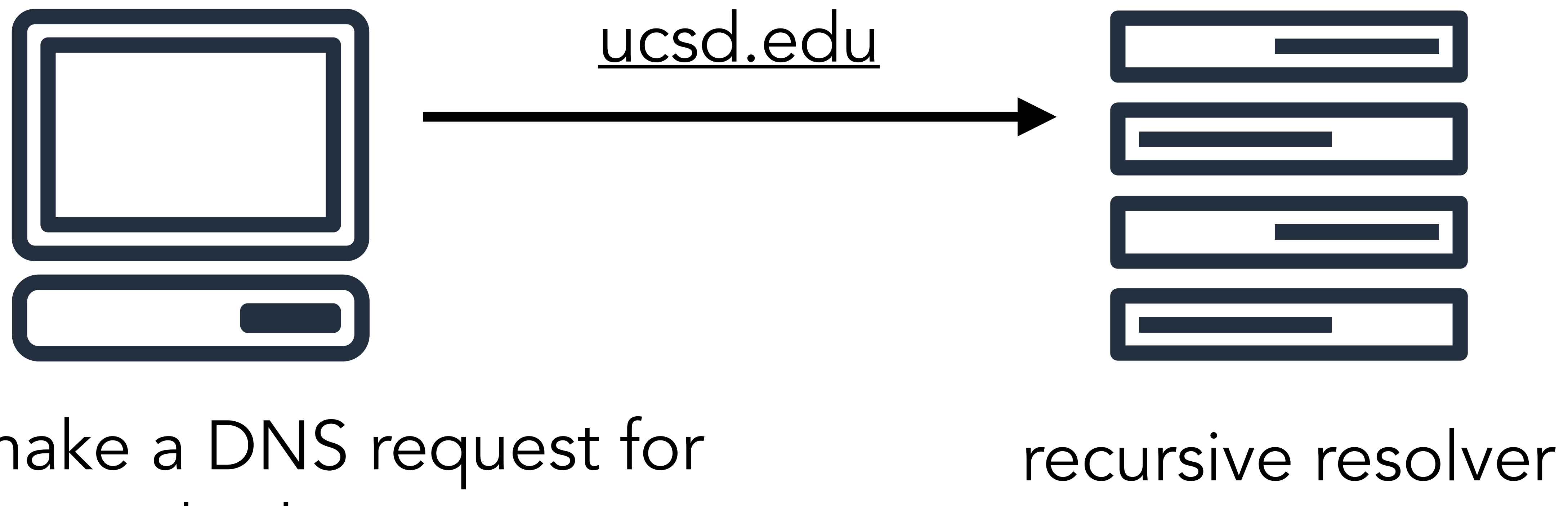
DNS Root	*13 root servers*	ICANN, iana Internet Assigned Numbers Authority
TLD	**.edu, .com,** etc.,	EDUCAUSE, VERISIGN™
Authoritative	**ucsd.edu**	UC San Diego
Authoritative	**deepak.ucsd.edu**	

Life of a DNS query

I want to make a DNS request for
ucsd.edu

Who do I talk to first?

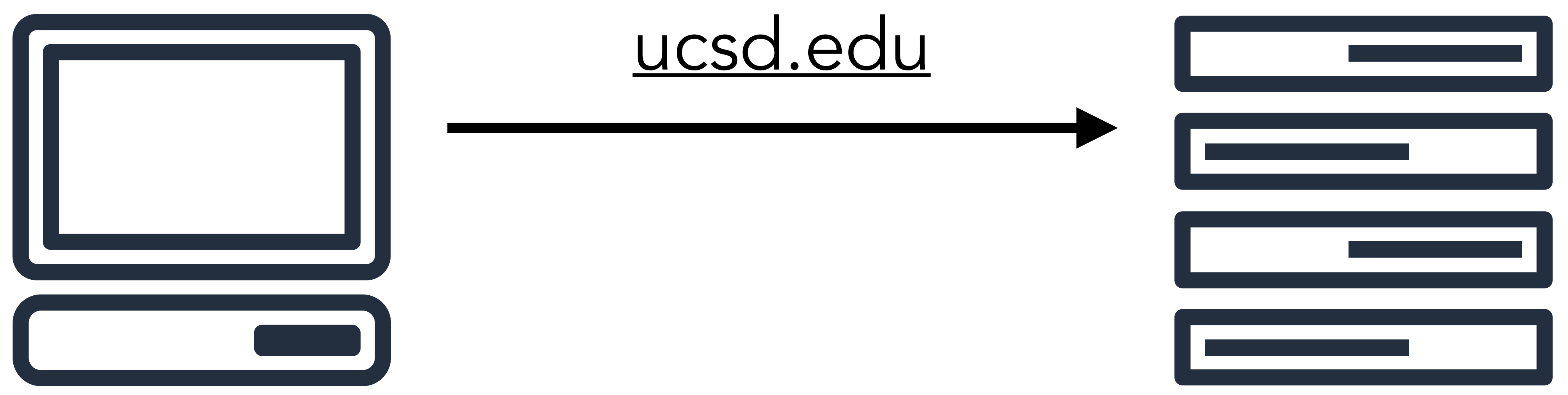
Life of a DNS query



I want to make a DNS request for ucsd.edu

Who do I talk to first?

Life of a DNS query

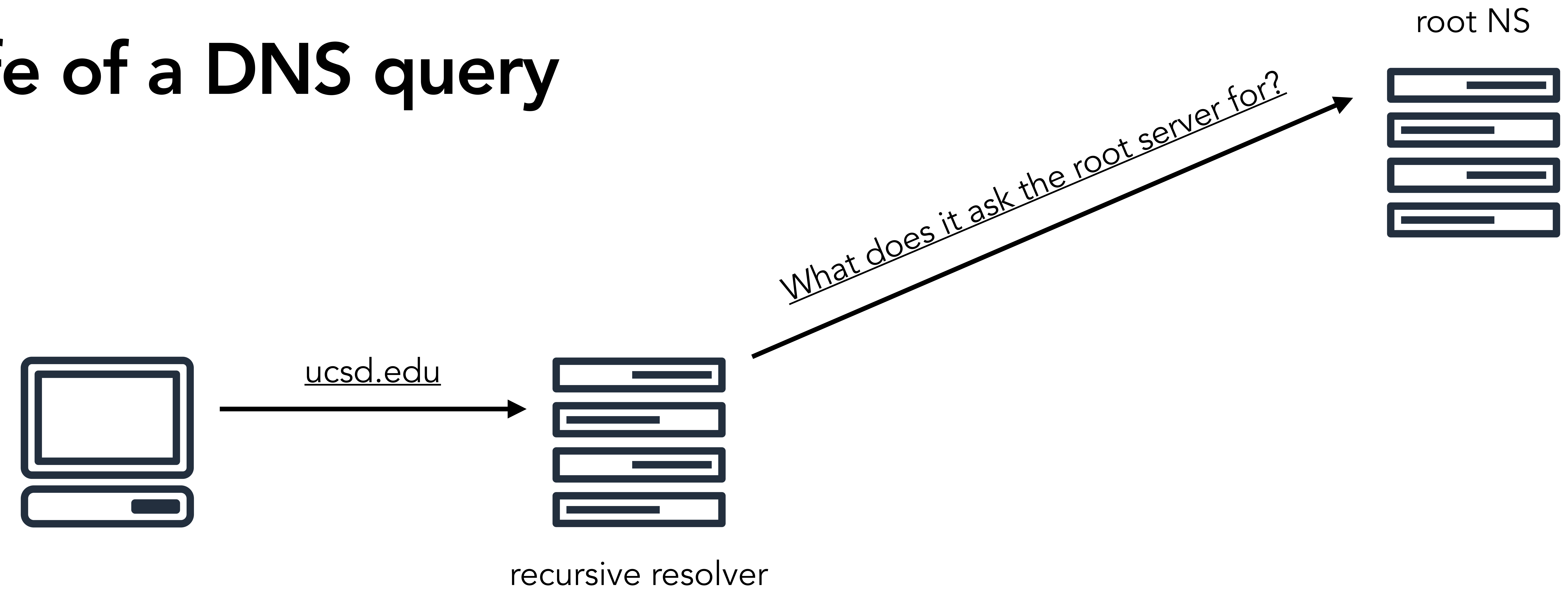


recursive resolver

The resolver has never heard of ucsd.edu.

Where does it go next?

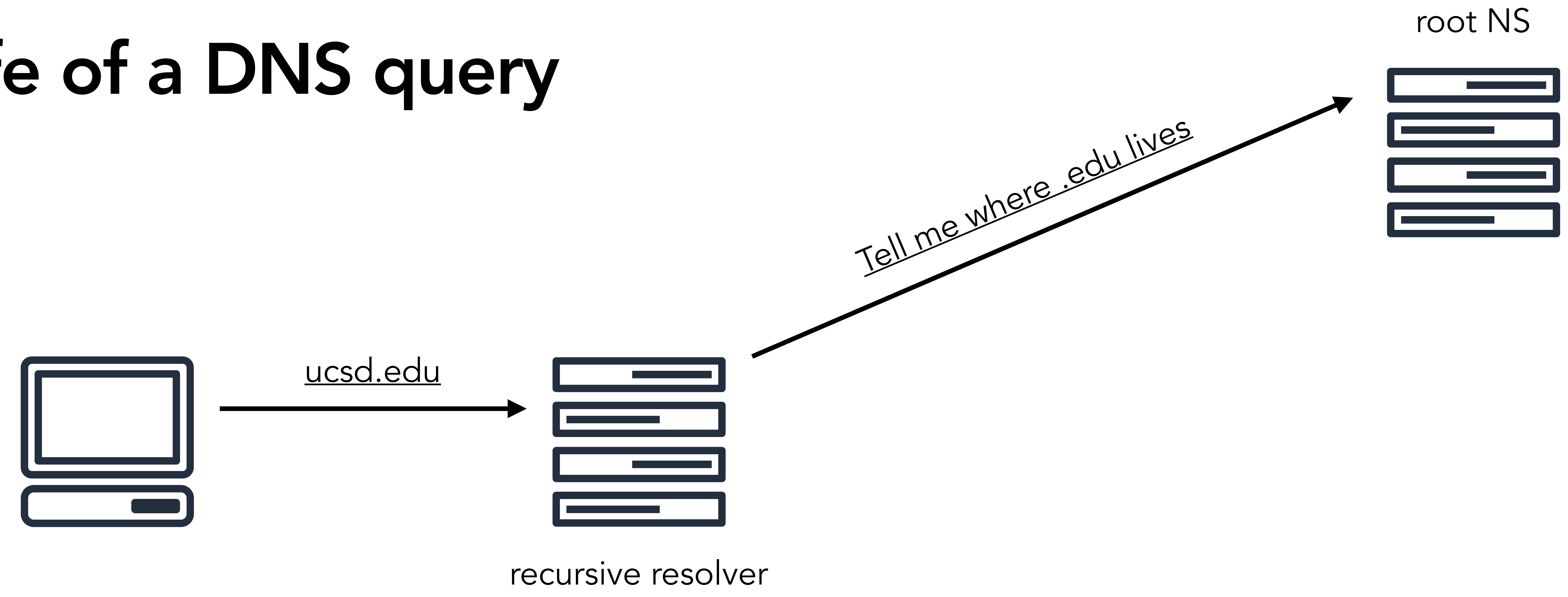
Life of a DNS query



The resolver has never heard of ucsd.edu.

Where does it go next?

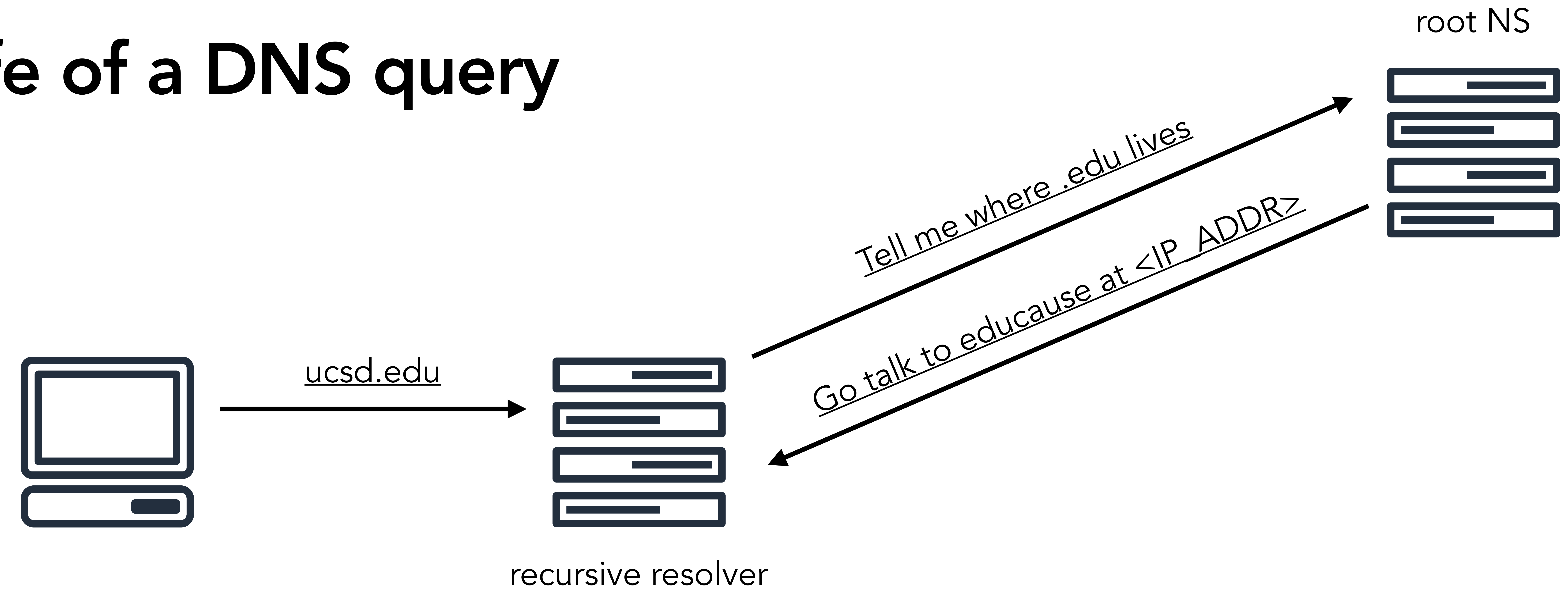
Life of a DNS query



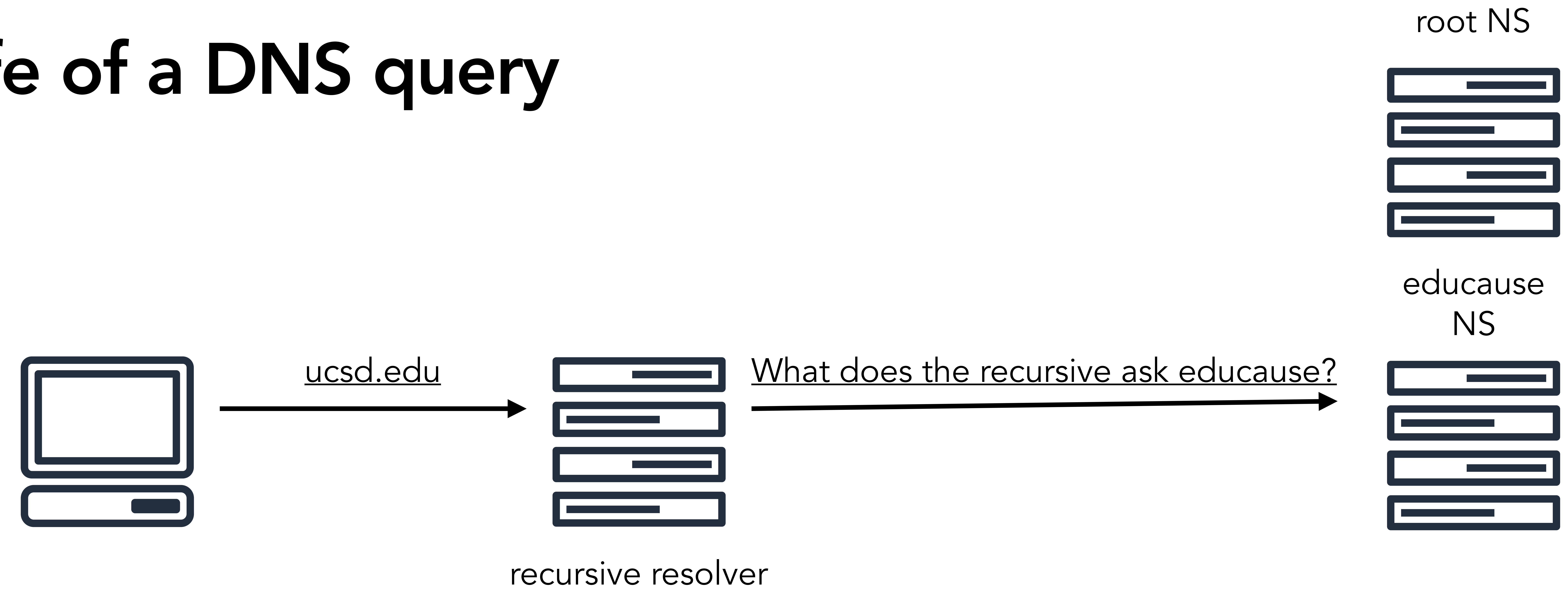
The resolver has never heard of ucsd.edu.

Where does it go next?

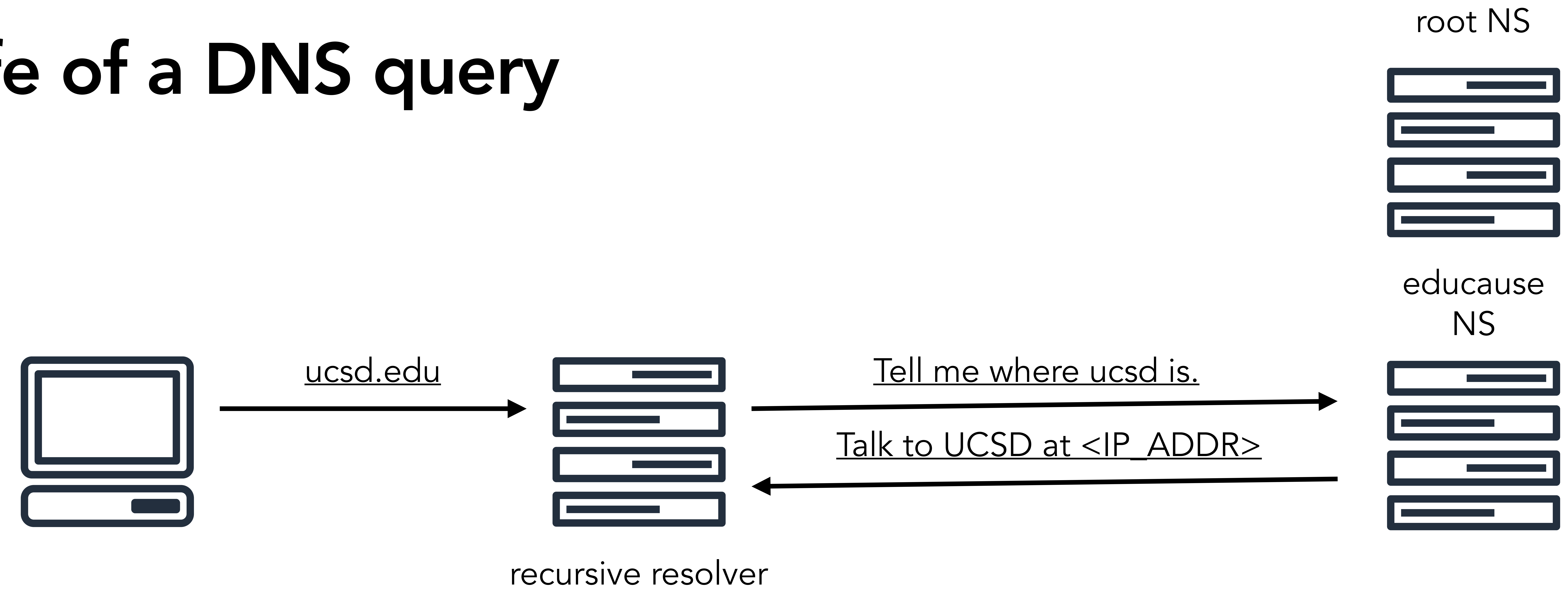
Life of a DNS query



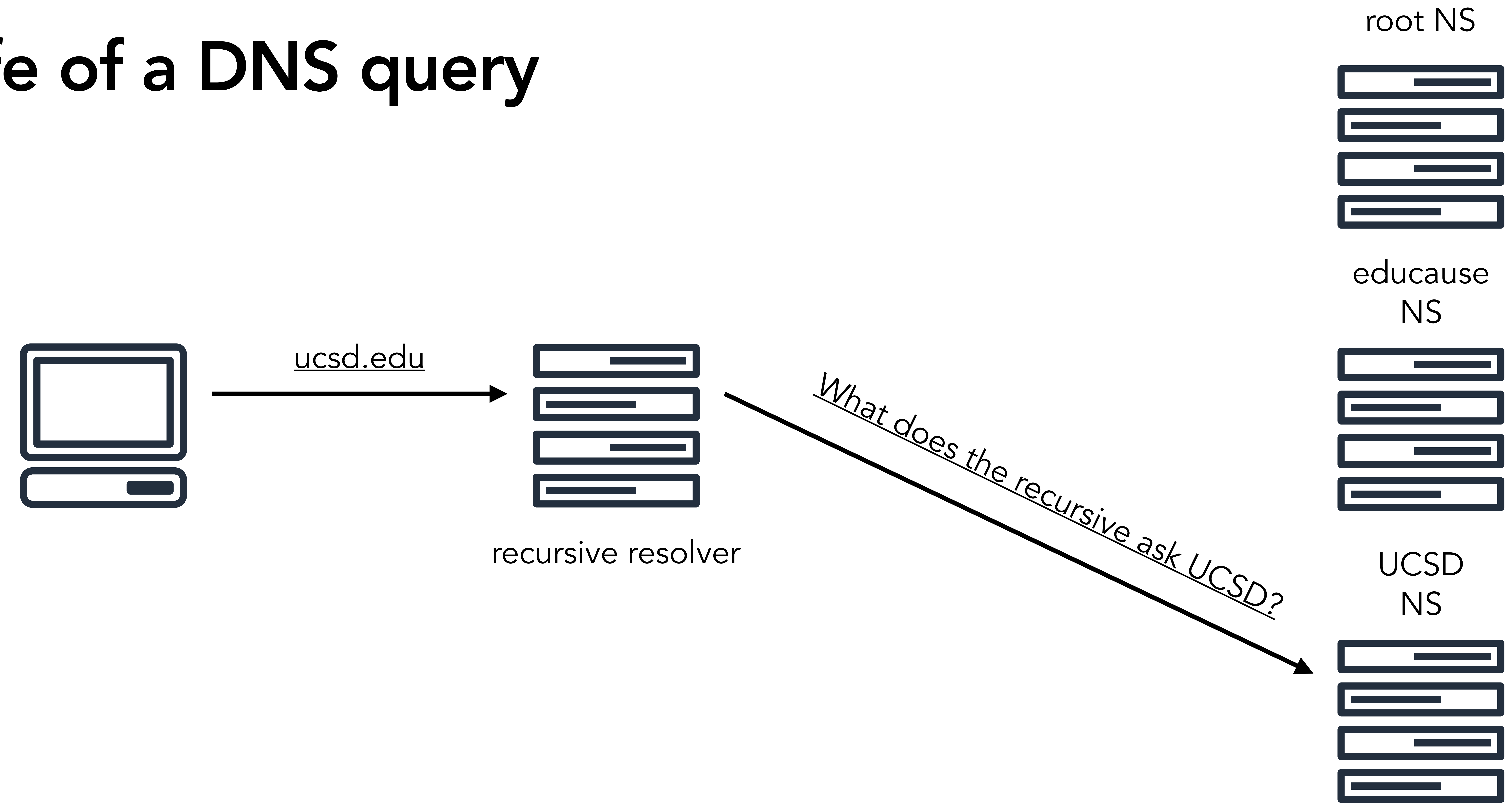
Life of a DNS query



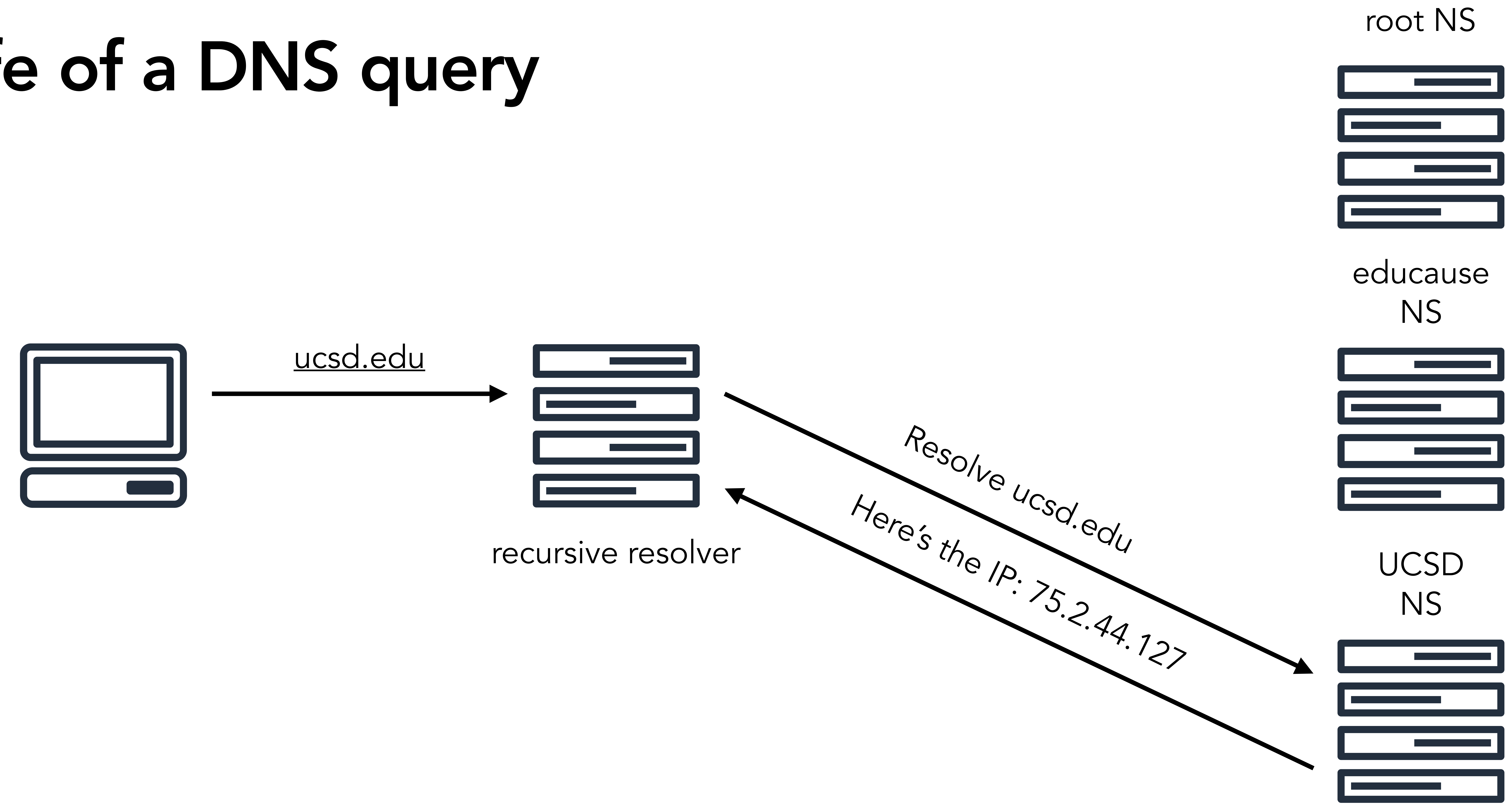
Life of a DNS query



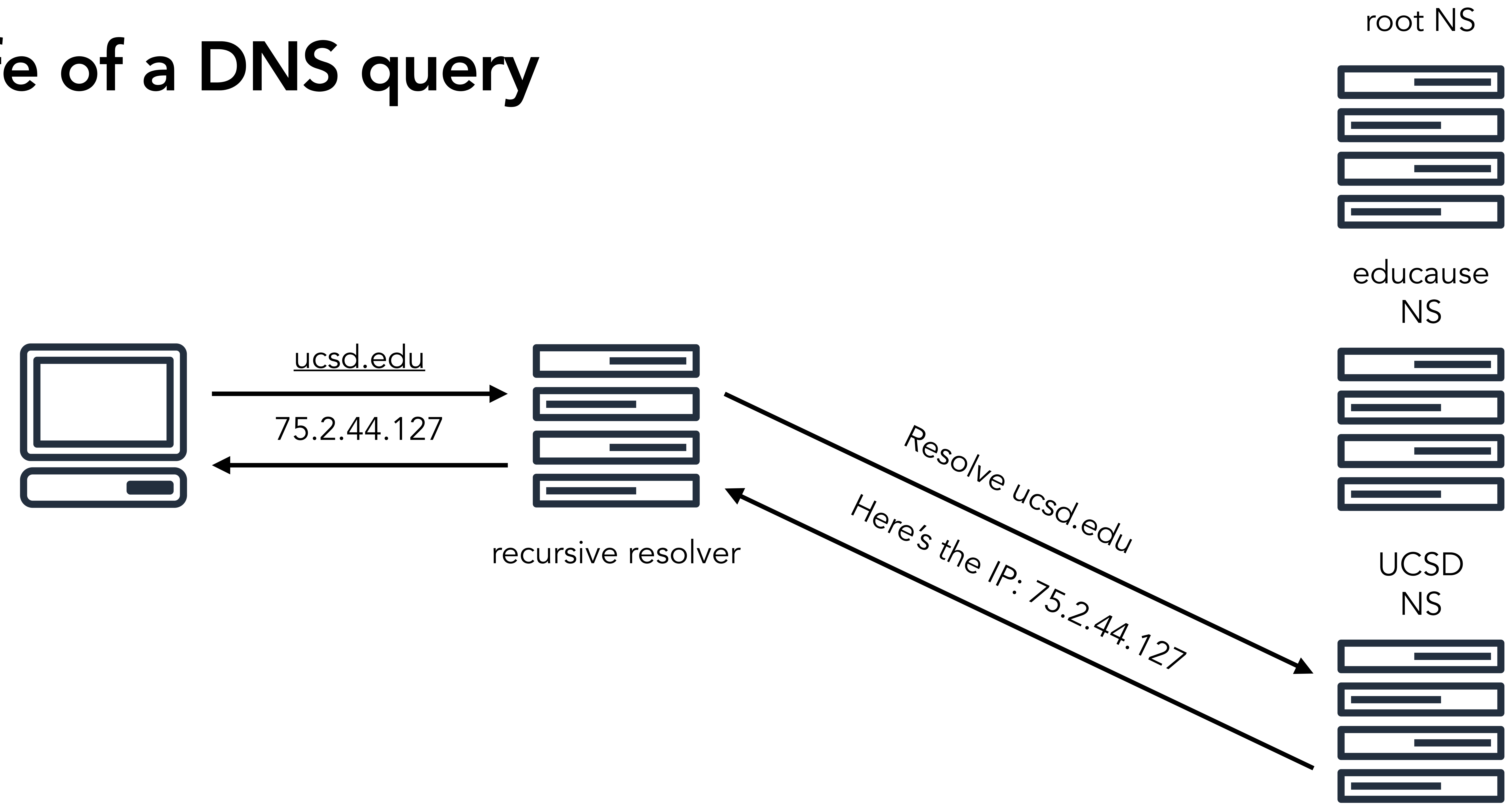
Life of a DNS query



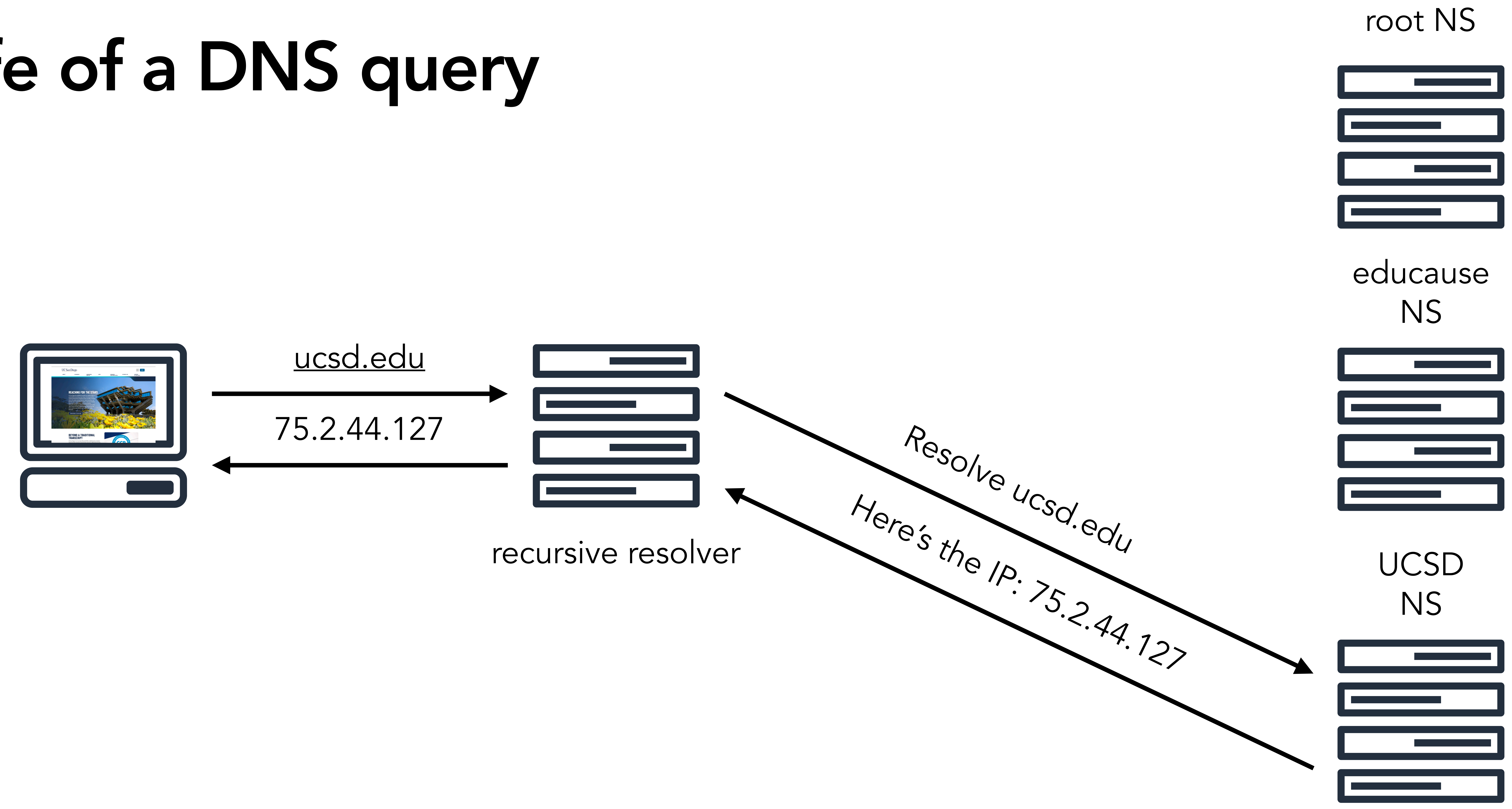
Life of a DNS query



Life of a DNS query



Life of a DNS query



DNS Query Types

Many ways to encode things in the DNS

- Field in the DNS query packet called QTYPE defines the the thing you’re looking for on the other end
    - What is an NS query?
    - What is an A query?
    - What is an AAAA query?
    - What is an M query?
    - What is a TXT query?

Do we always need to hit a root server to get an answer?

Do we always need to hit a root server to get an answer?

No! We have DNS *caches* for this purpose. What's a DNS cache?

DNS Caching

Making things fast

- Where do DNS records get cached? Who caches them?

DNS Caching

Making things fast

- Where do DNS records get cached? Who caches them?
  - The browser has a DNS cache, your OS has its own DNS cache, and the recursive also has a DNS cache

DNS Caching

Making things fast

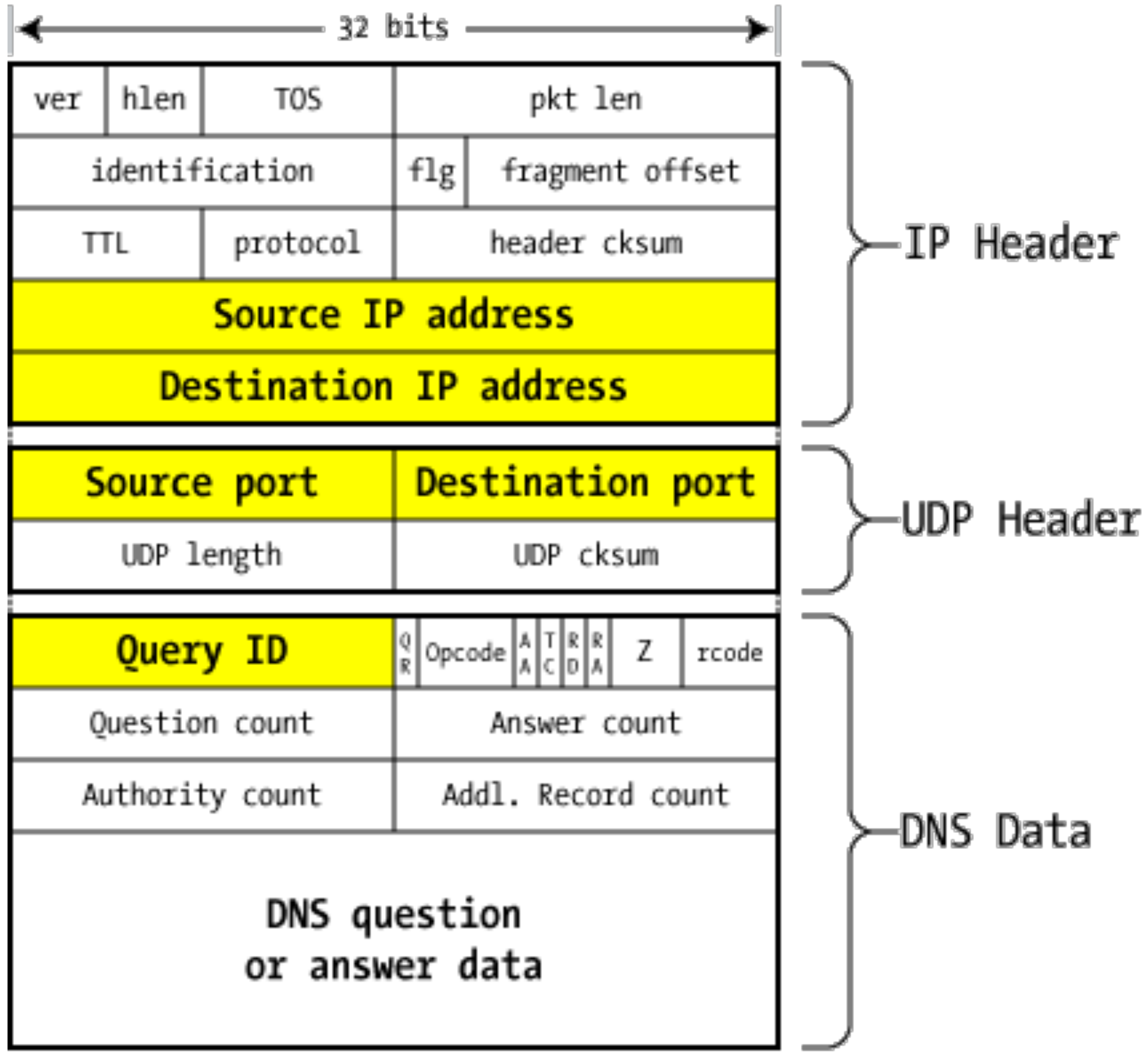
- Where do DNS records get cached? Who caches them?
  - The browser has a DNS cache, your OS has its own DNS cache, and the recursive also has a DNS cache
- What is the time-to-live (TTL)? How does this affect caching?

DNS Caching

Making things fast

- Where do DNS records get cached? Who caches them?
    - The browser has a DNS cache, your OS has its own DNS cache, and the recursive also has a DNS cache
- What is the time-to-live (TTL)? How does this affect caching?
- Thought experiment: How frequently are the root servers contacted?

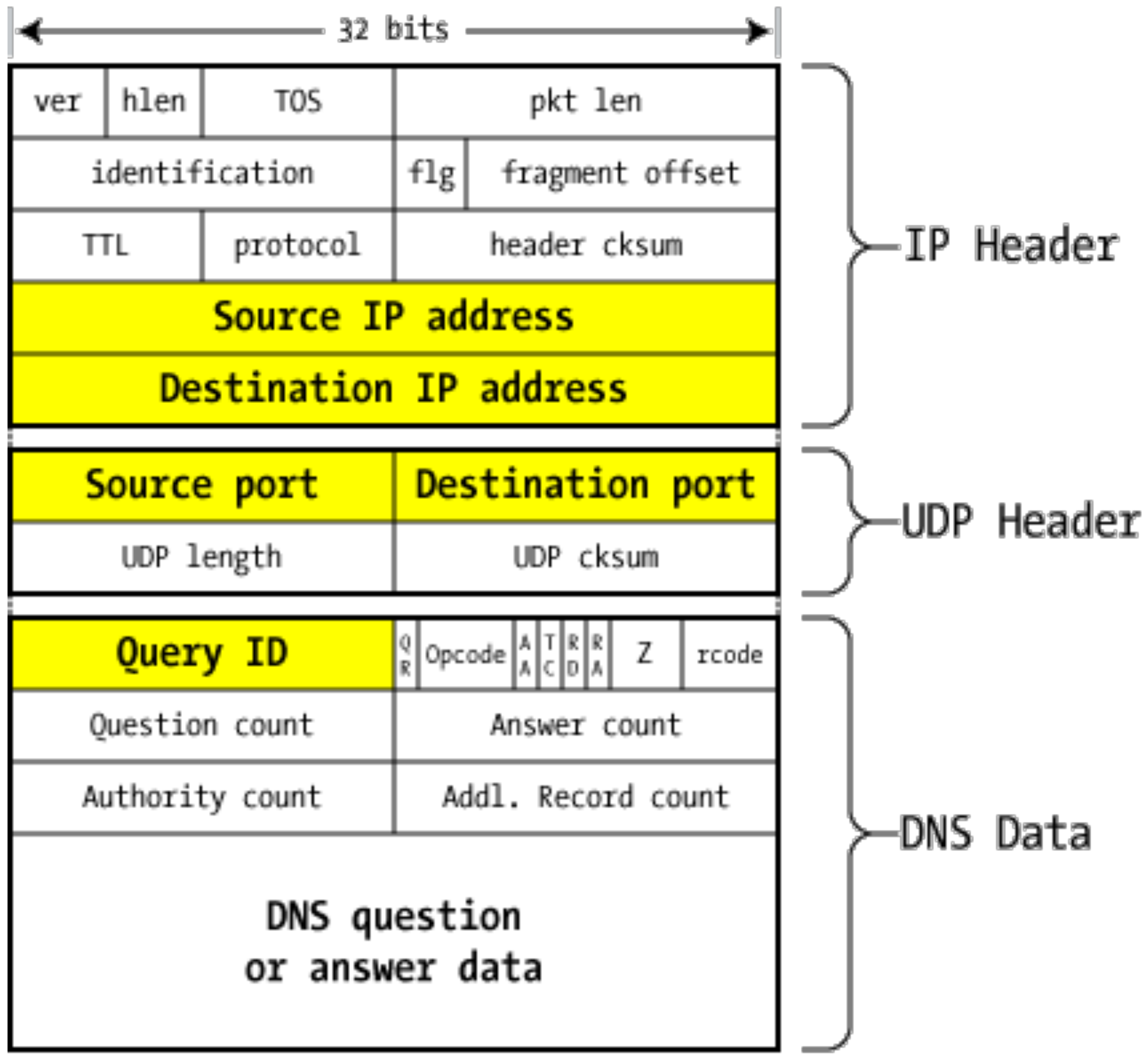
Unpacking DNS Packets



DNS packet on the wire

- DNS typically operates over UDP over IP
  - What is UDP? How is it different from TCP?

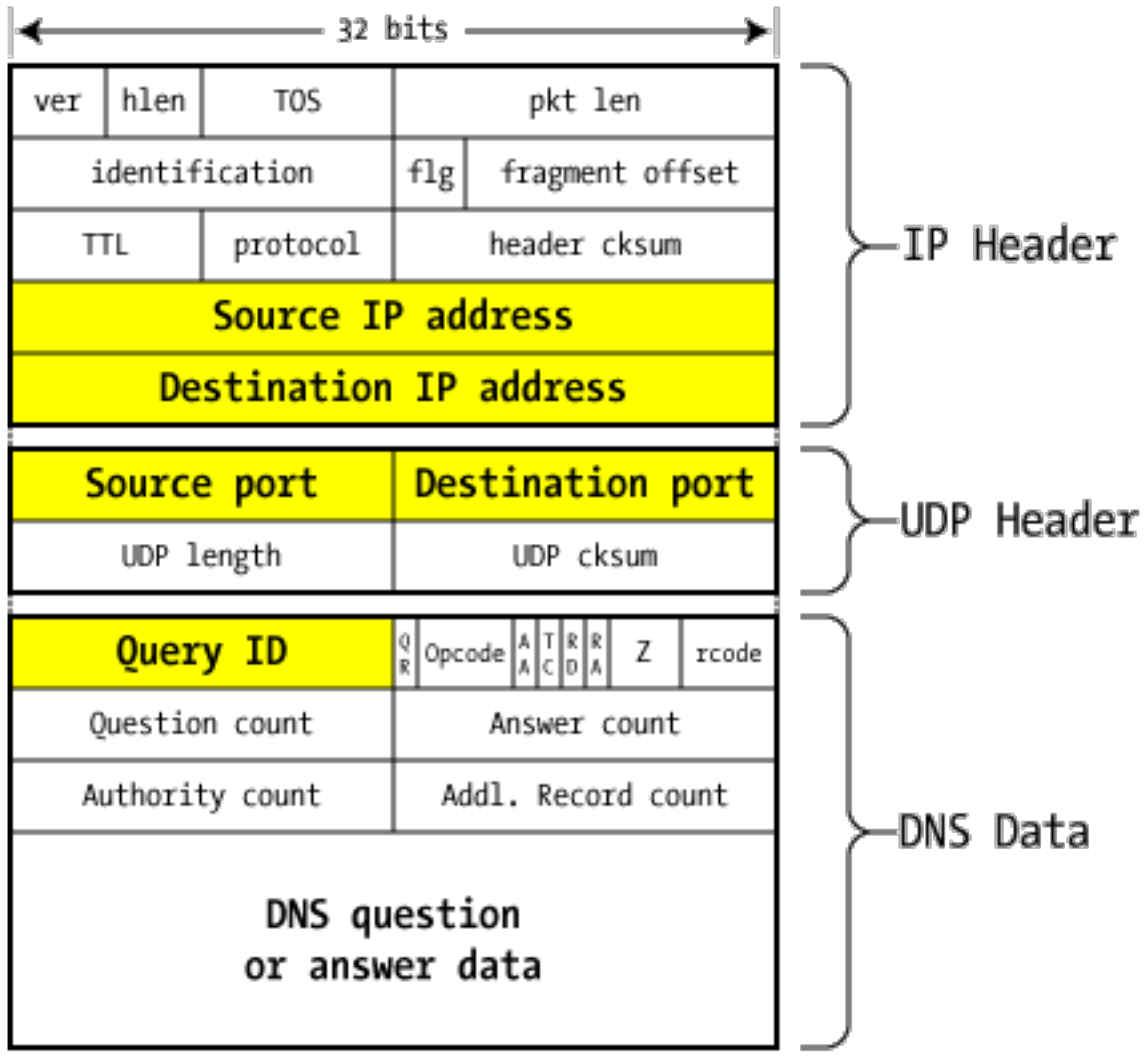
Unpacking DNS Packets



DNS packet on the wire

- DNS typically operates over UDP over IP
  - What is UDP? How is it different from TCP?
- What are ports? What's a source port and destination port?

Unpacking DNS Packets



DNS packet on the wire

- DNS typically operates over UDP over IP
    - What is UDP? How is it different from TCP?
- What are ports? What's a source port and destination port?
- What's a query ID? How big is it?

Break Time + Attendance

Codeword:
Complex-Domains

https://tinyurl.com/cse227-attend

DNS Cache Poisoning

A few words on this vuln…

- Released in 2008, well after DNS was established for many years
- Affected almost every single DNS recursor / cache on the planet: which made it *very* important to fix quickly
- While we're talking: think about what this means for *trust* on the Internet

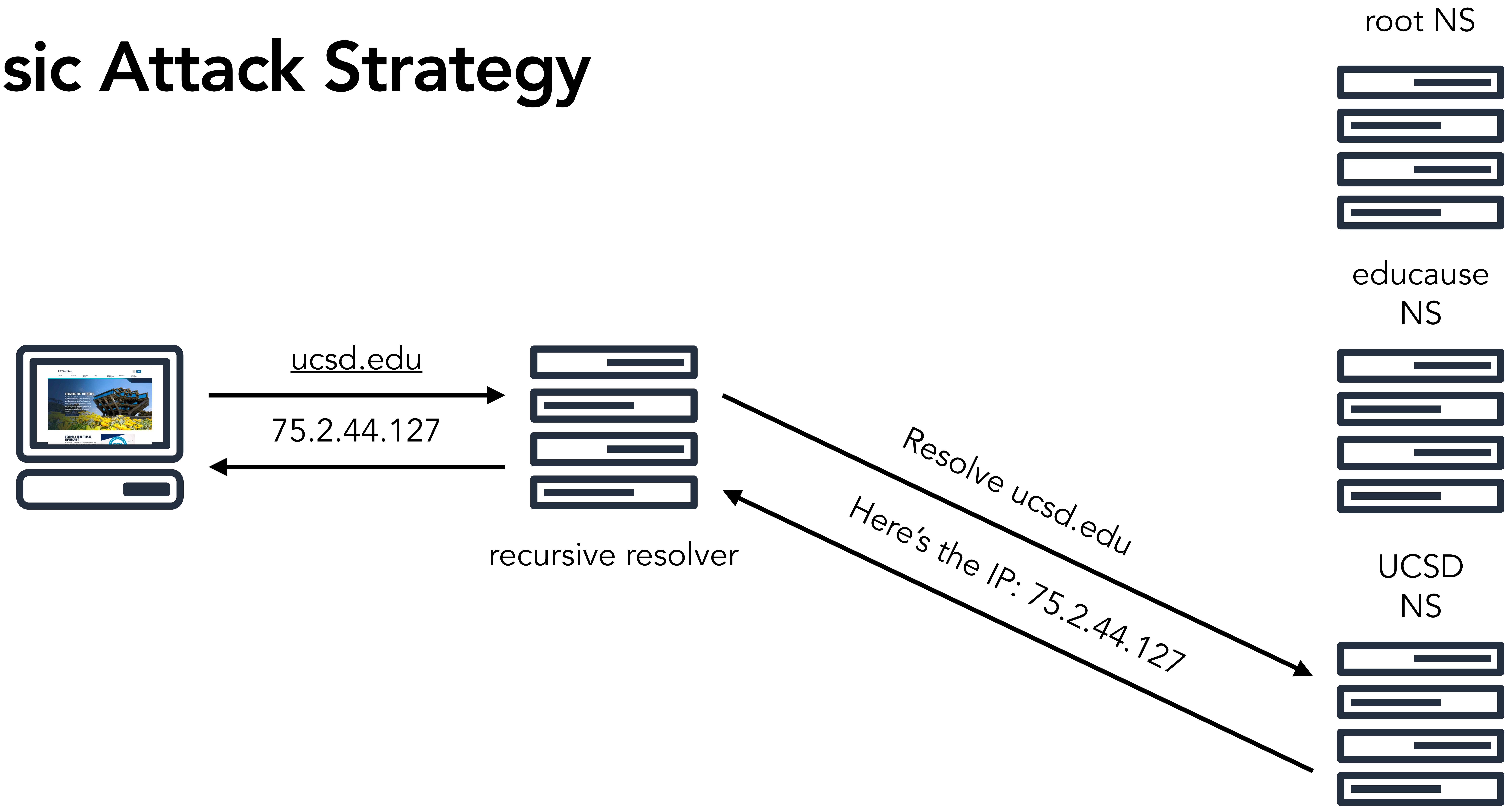
Basic Premise of the Attack

- What is the goal of an attacker in the DNS cache poisoning attack?

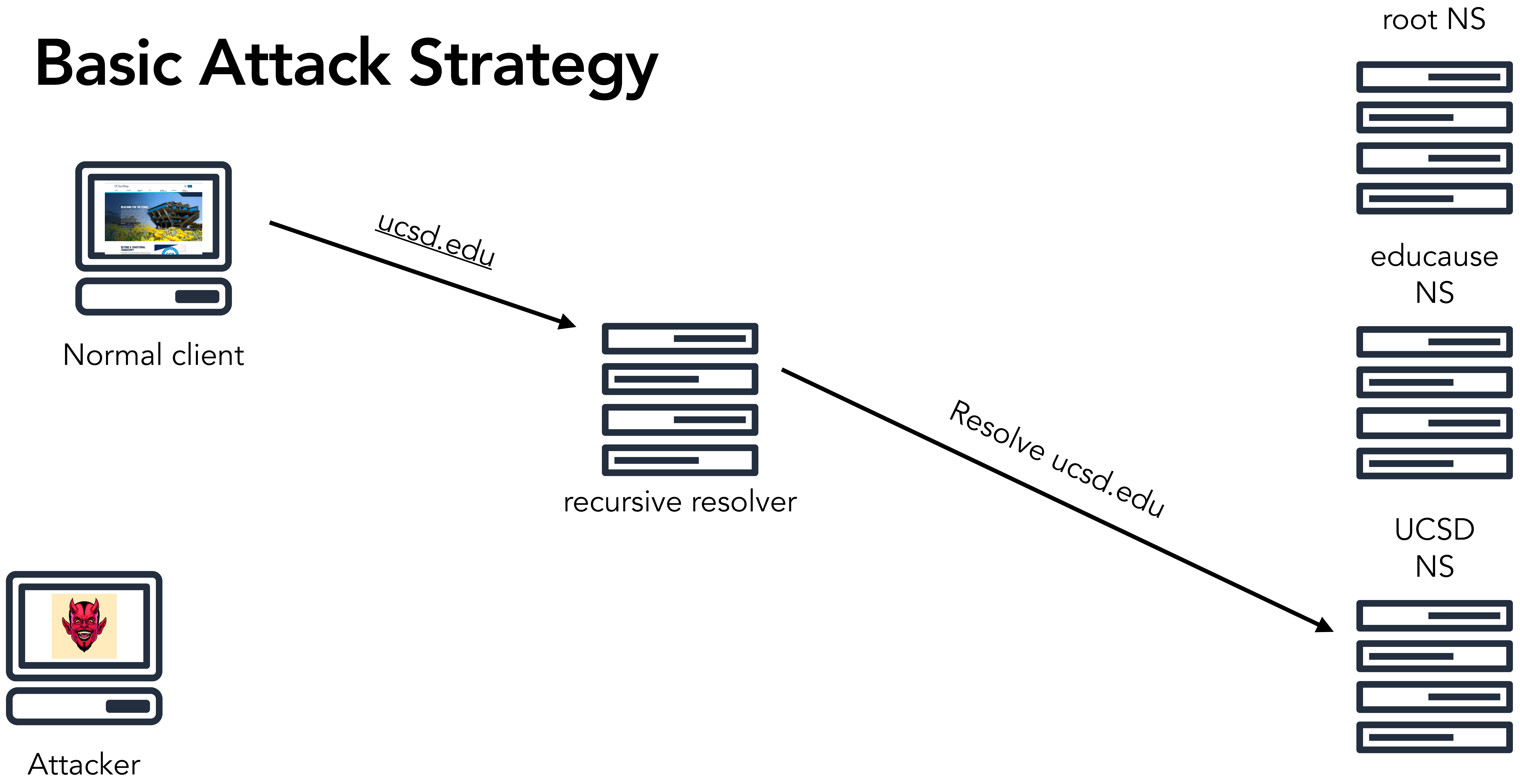
Basic Premise of the Attack

- What is the goal of an attacker in the DNS cache poisoning attack?
  - Goal: Get a record inside of a DNS cache that is **wrong,** tricking clients transparently
  - Simple version: Record for a single hostname
  - Kaminsky version: NS record for an *entire zone*

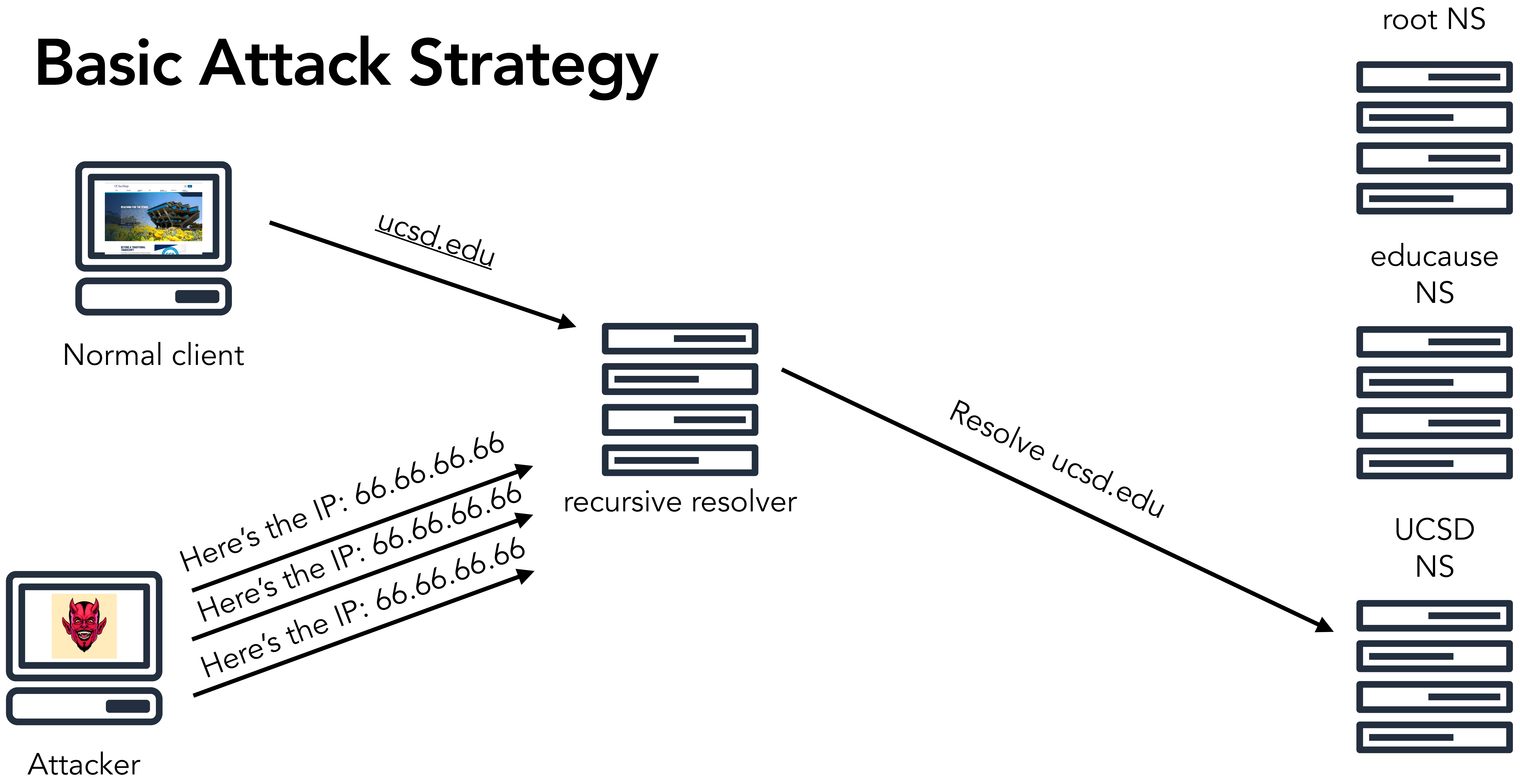
Basic Attack Strategy



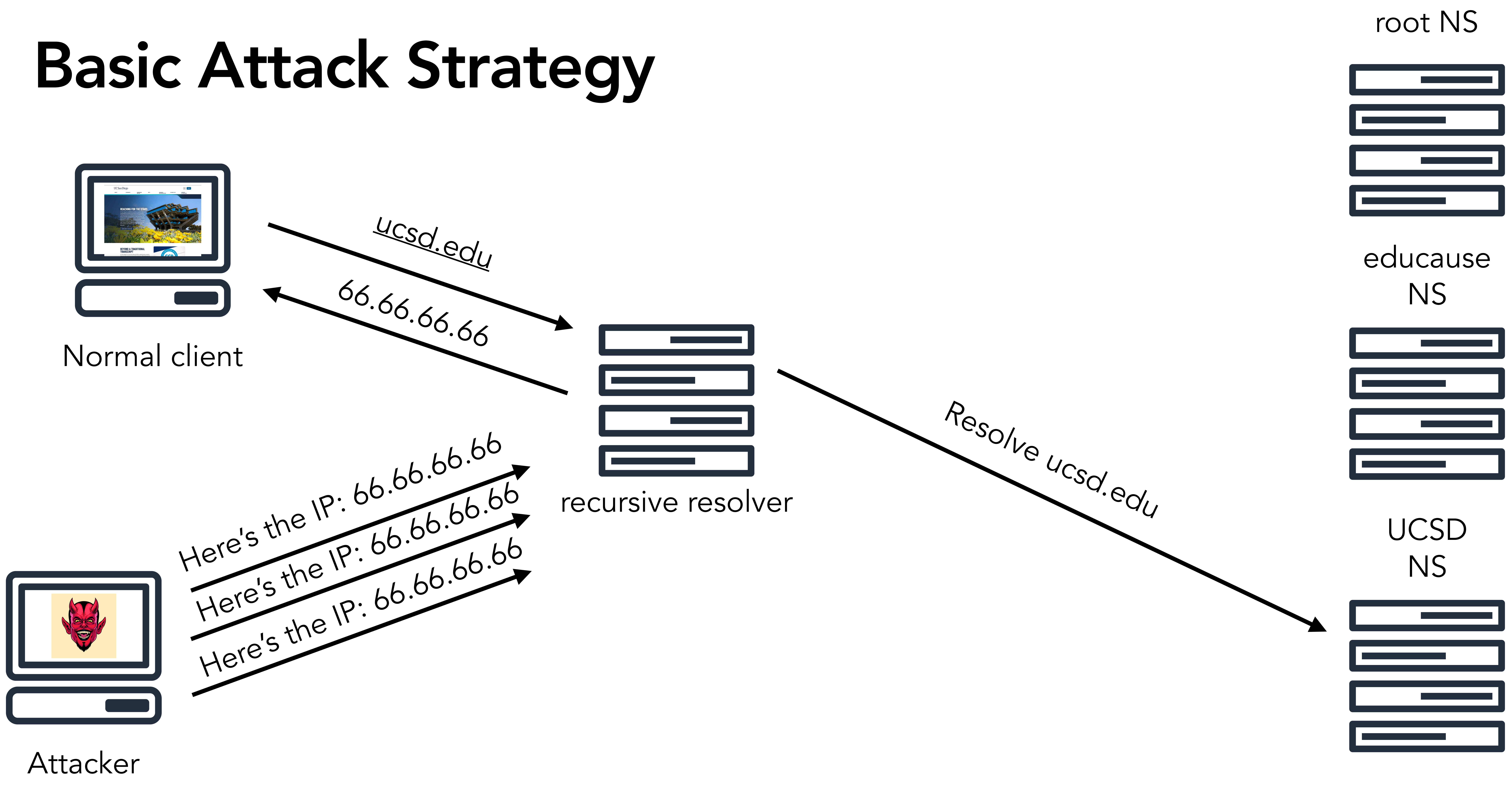
Basic Attack Strategy



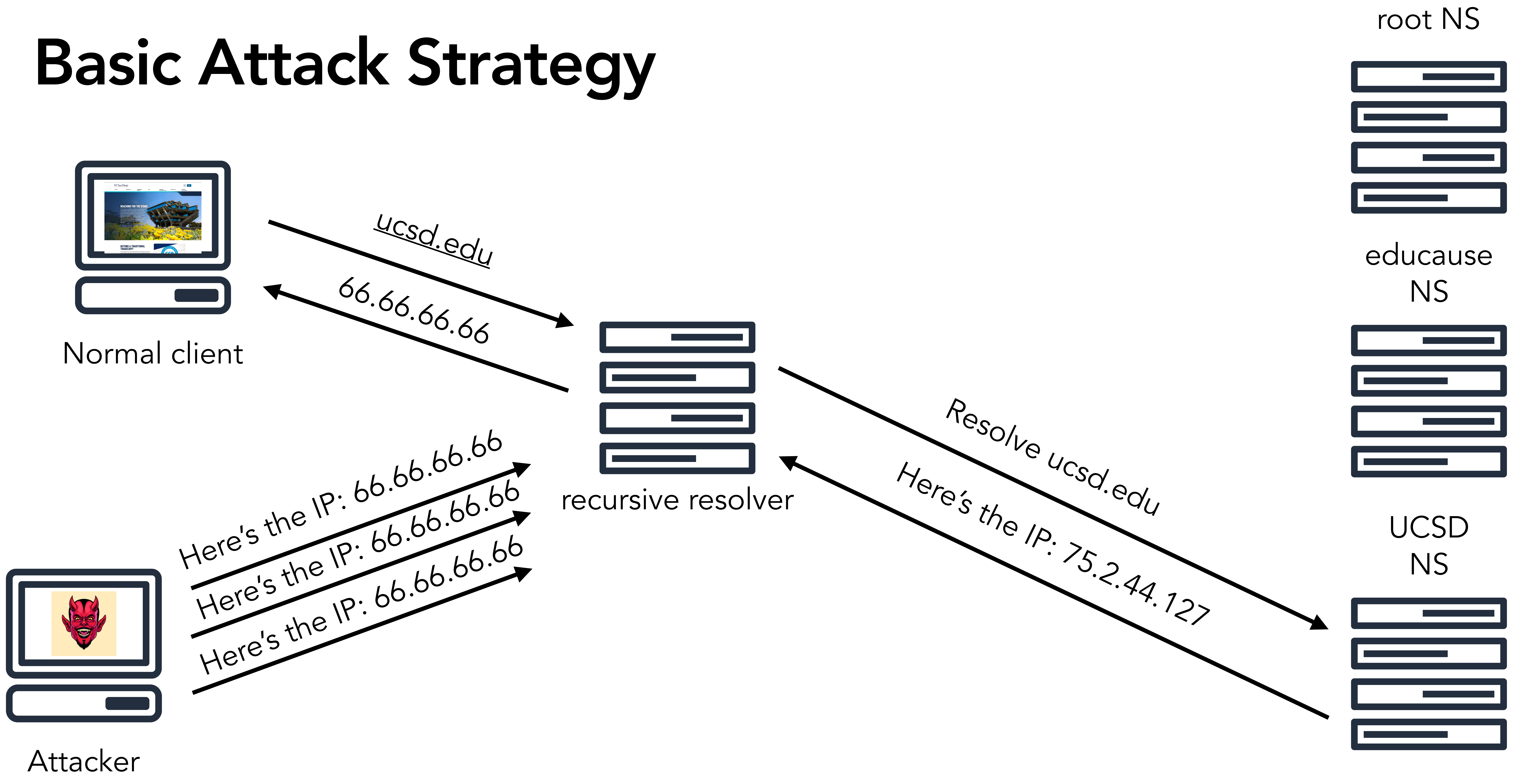
Basic Attack Strategy



Basic Attack Strategy

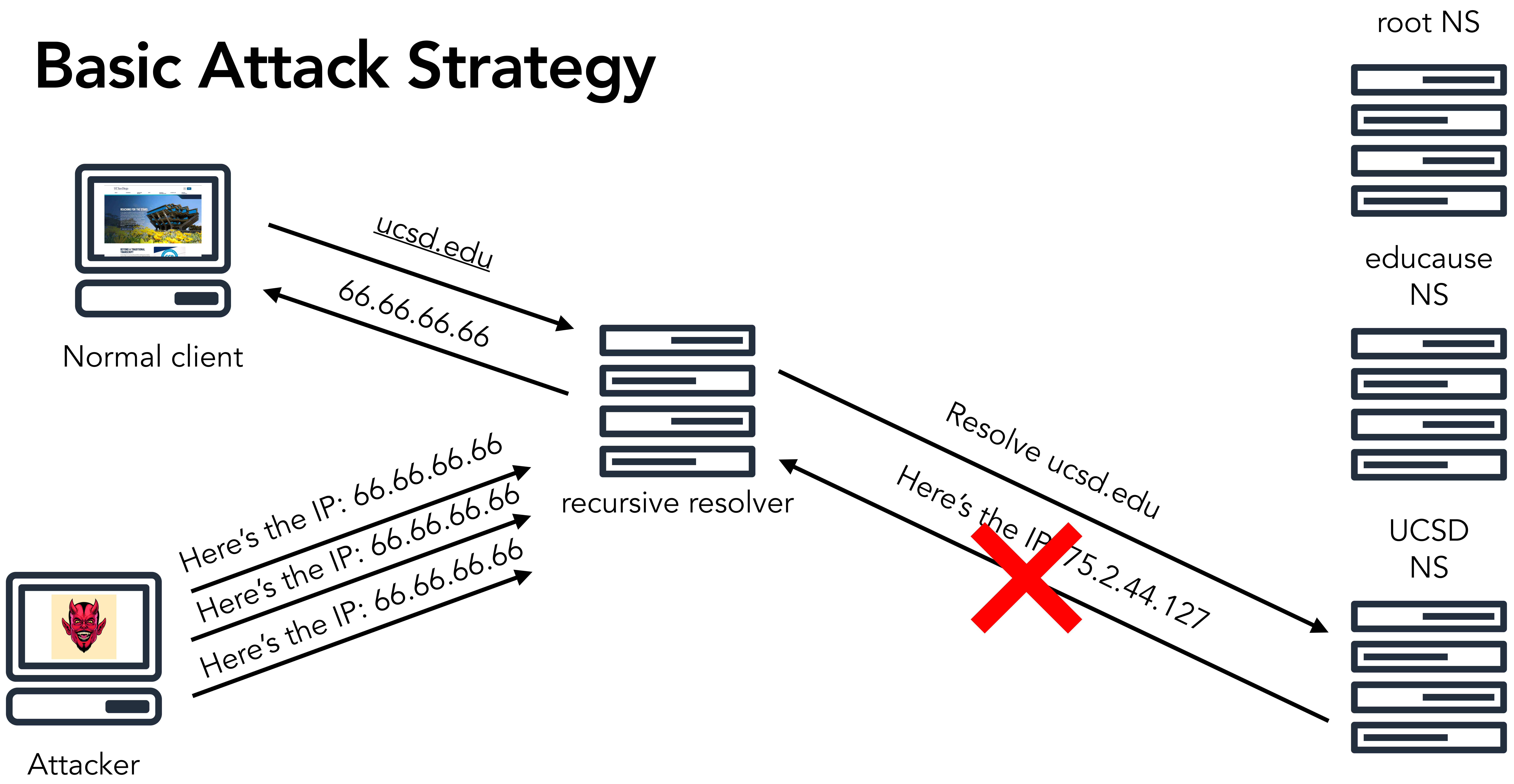


Basic Attack Strategy



root NS
educause NS
UCSD NS
Normal client
ucsd.edu
66.66.66.66
recursive resolver
Resolve ucsd.edu
Here's the IP: 75.2.44.127
Here's the IP: 66.66.66.66
Here's the IP: 66.66.66.66
Here's the IP: 66.66.66.66
Attacker

Basic Attack Strategy


root NS
educause NS
UCSD NS
Normal client
ucsd.edu
66.66.66.66
recursive resolver
Resolve ucsd.edu
Here's the IP: 75.2.44.127
Here's the IP: 66.66.66.66
Here's the IP: 66.66.66.66
Here's the IP: 66.66.66.66
Attacker

Making it work in practice

- How does the recursive resolver know to trust a response from a NS?

Making it work in practice

- How does the recursive resolver know to trust a response from a NS?
    - Query ID needs to match its outgoing query ID
    - Name it's responding to needs to be the same
    - UDP fields need to be the same, chiefly SRC_IP and SRC_Port

Making it work in practice

- How does the recursive resolver know to trust a response from a NS?
    - Query ID needs to match its outgoing query ID
    - Name it’s responding to needs to be the same
    - UDP fields need to be the same, chiefly SRC_IP and SRC_Port
- How does the recursive resolver know that the NS responding the query actually owns the name it’s claiming to know about?

Making it work in practice

- How does the recursive resolver know to trust a response from a NS?
  - Query ID needs to match its outgoing query ID
  - Name it’s responding to needs to be the same
  - UDP fields need to be the same, chiefly SRC_IP and SRC_Port
- How does the recursive resolver know that the NS responding the query actually owns the name it’s claiming to know about?
  - It *doesn’t*. Anyone can be authoritative for any name!

Back in ’08…

- Attacker needs to spoof NS responses
  - How did the attacker learn the Query ID?
  - How did the attacker learn the UDP port used?

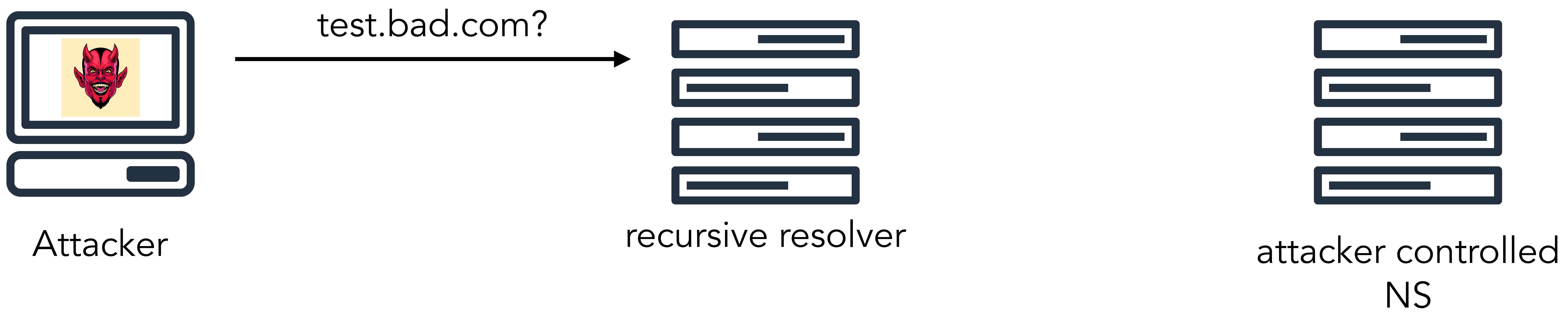
Back in '08…

- Attacker needs to spoof NS responses
  - How did the attacker learn the Query ID?
  - How did the attacker learn the UDP port used?

Attacker
recursive resolver
attacker controlled NS

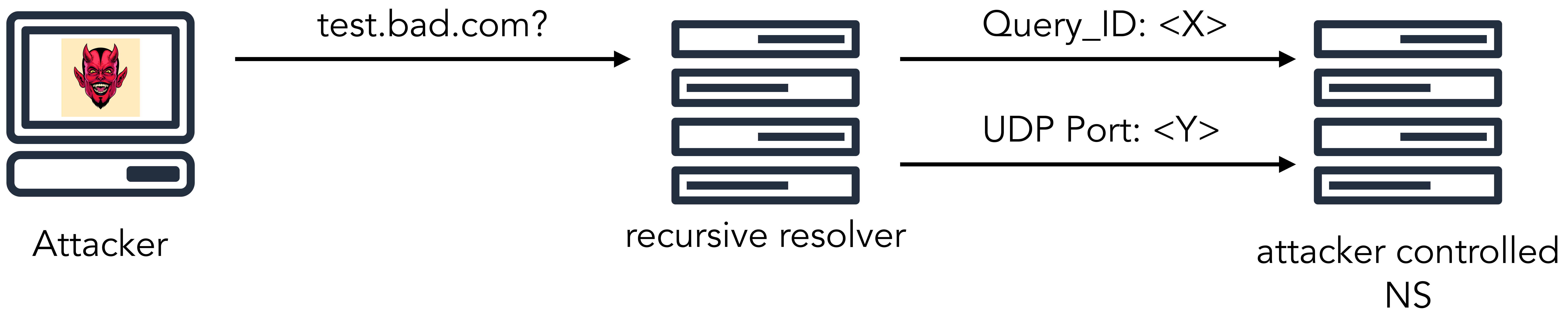
Back in '08…

- Attacker needs to spoof NS responses
  - How did the attacker learn the Query ID?
  - How did the attacker learn the UDP port used?



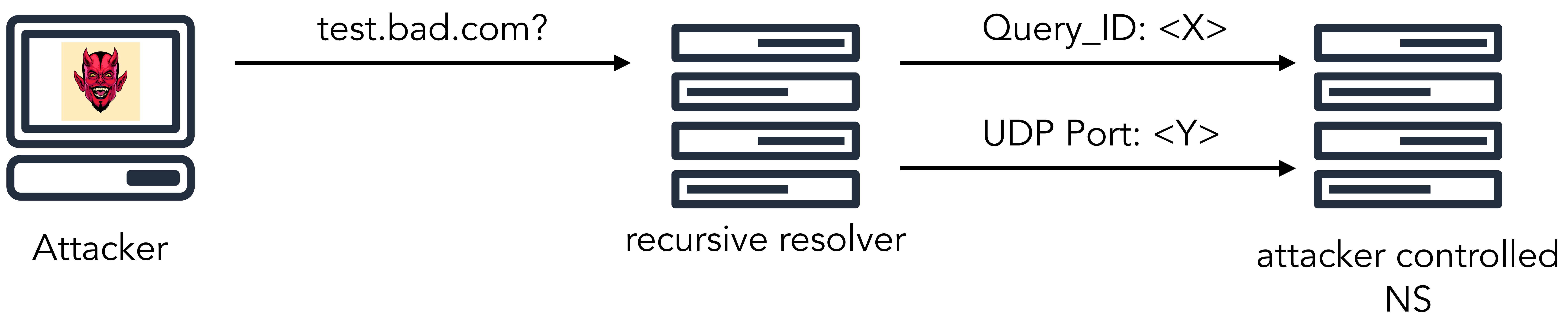
Back in '08…

- Attacker needs to spoof NS responses
  - How did the attacker learn the Query ID?
  - How did the attacker learn the UDP port used?



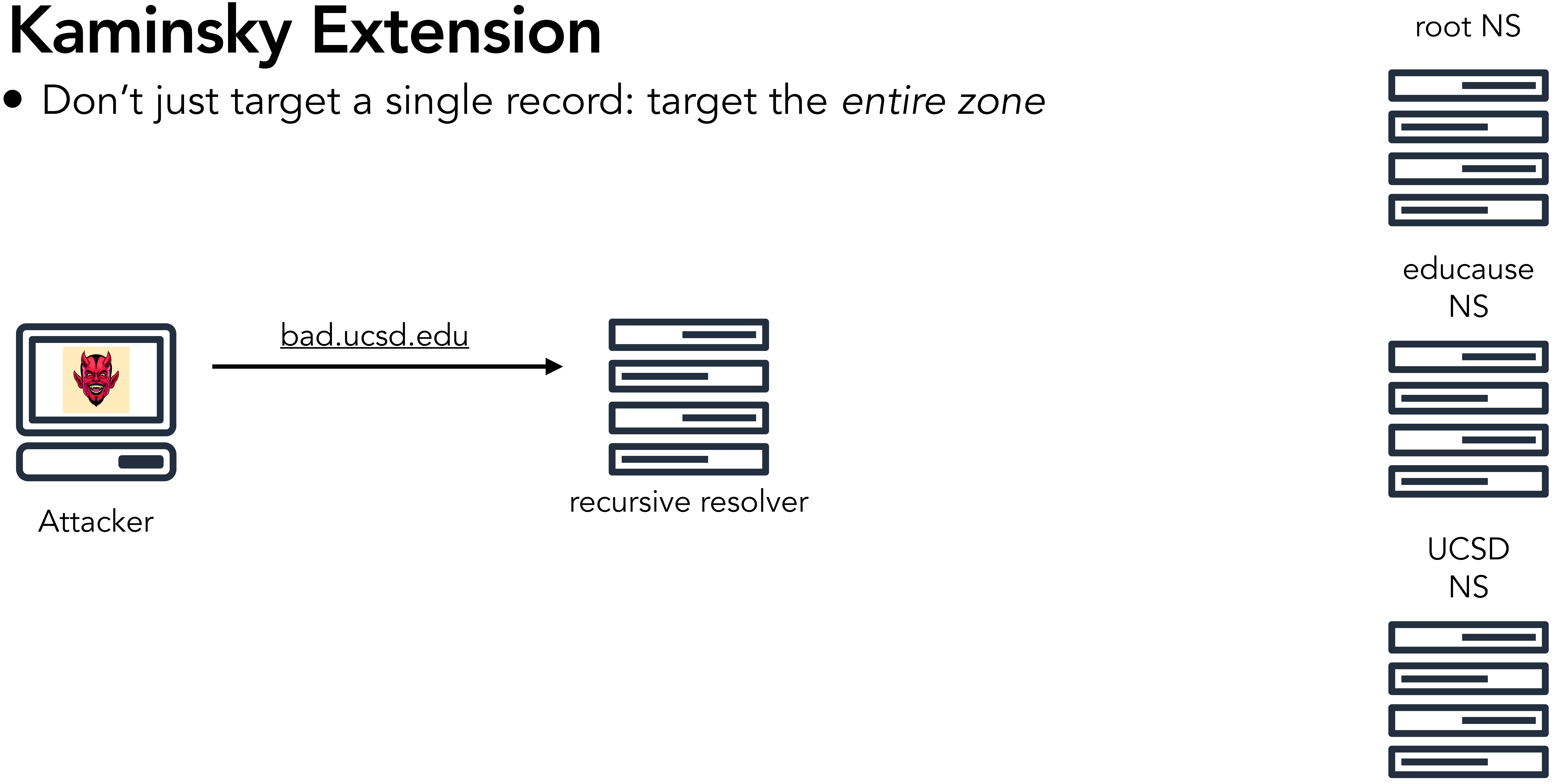
Back in ’08…

- Attacker needs to spoof NS responses
  - How did the attacker learn the Query ID? **Global, monotonically increasing**
  - How did the attacker learn the UDP port used? **Fixed for all DNS queries**



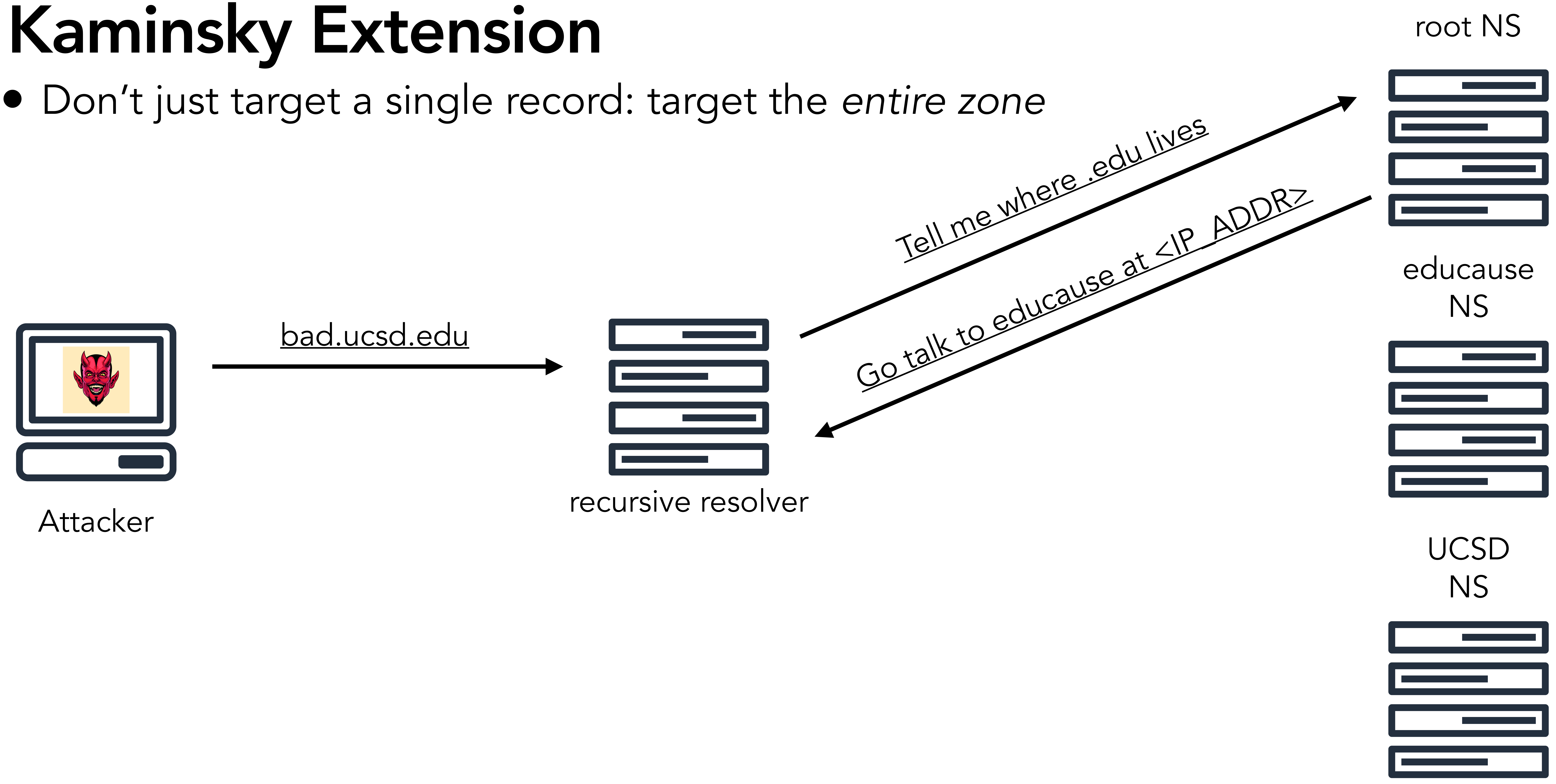
Kaminsky Extension

- Don’t just target a single record: target the *entire zone*



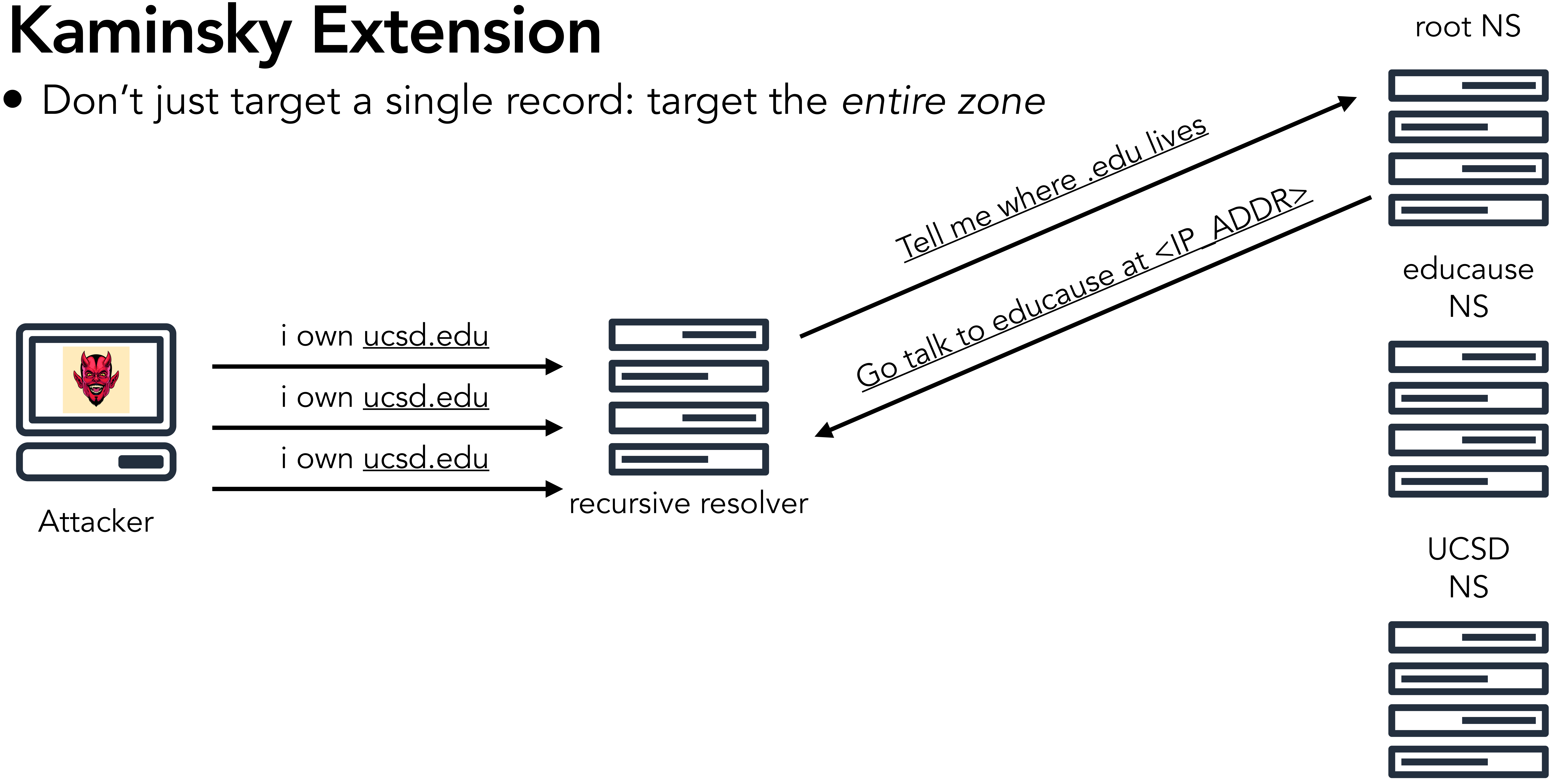
Kaminsky Extension

- Don’t just target a single record: target the *entire zone*



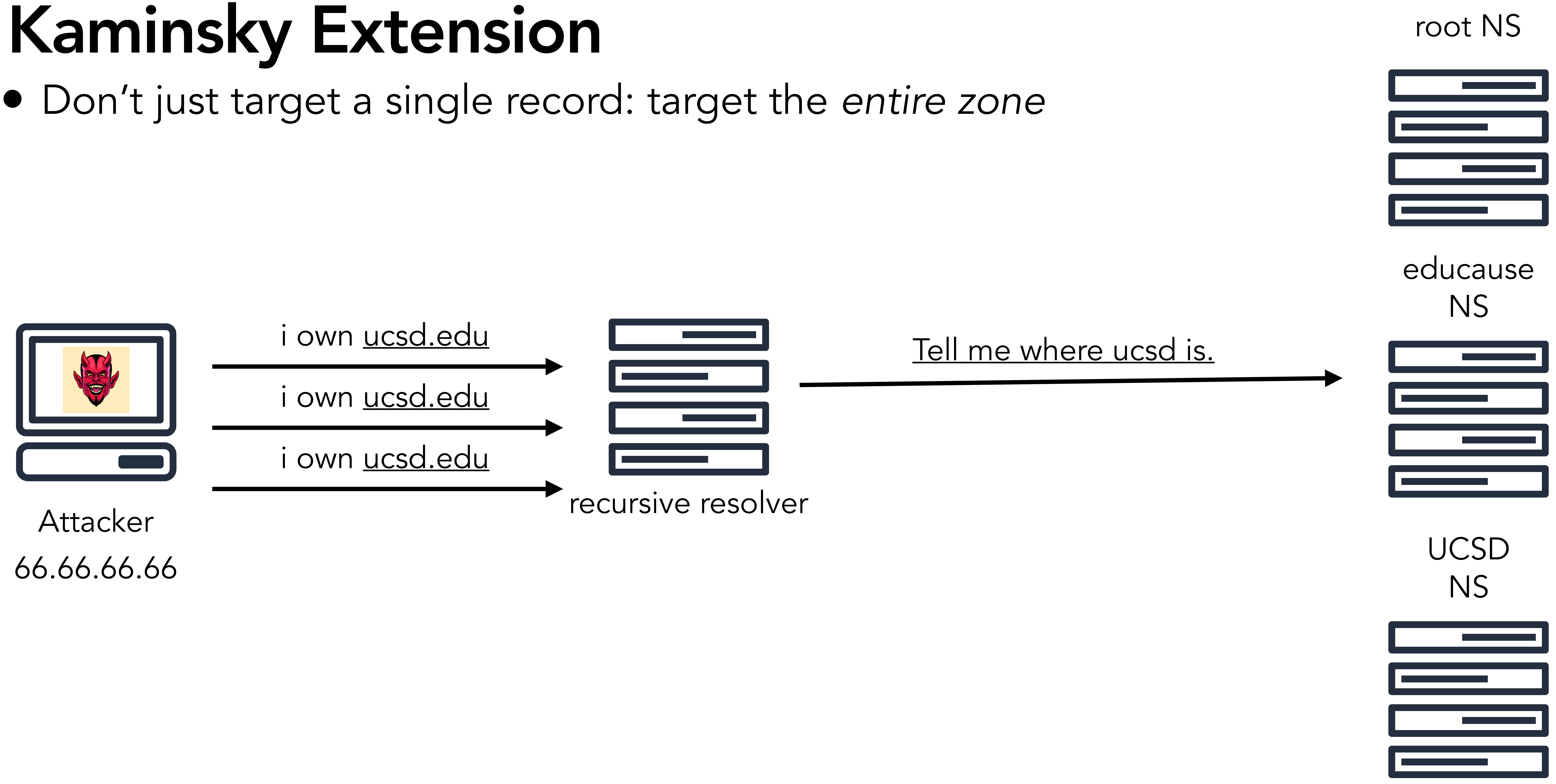
Kaminsky Extension

- Don't just target a single record: target the *entire zone*



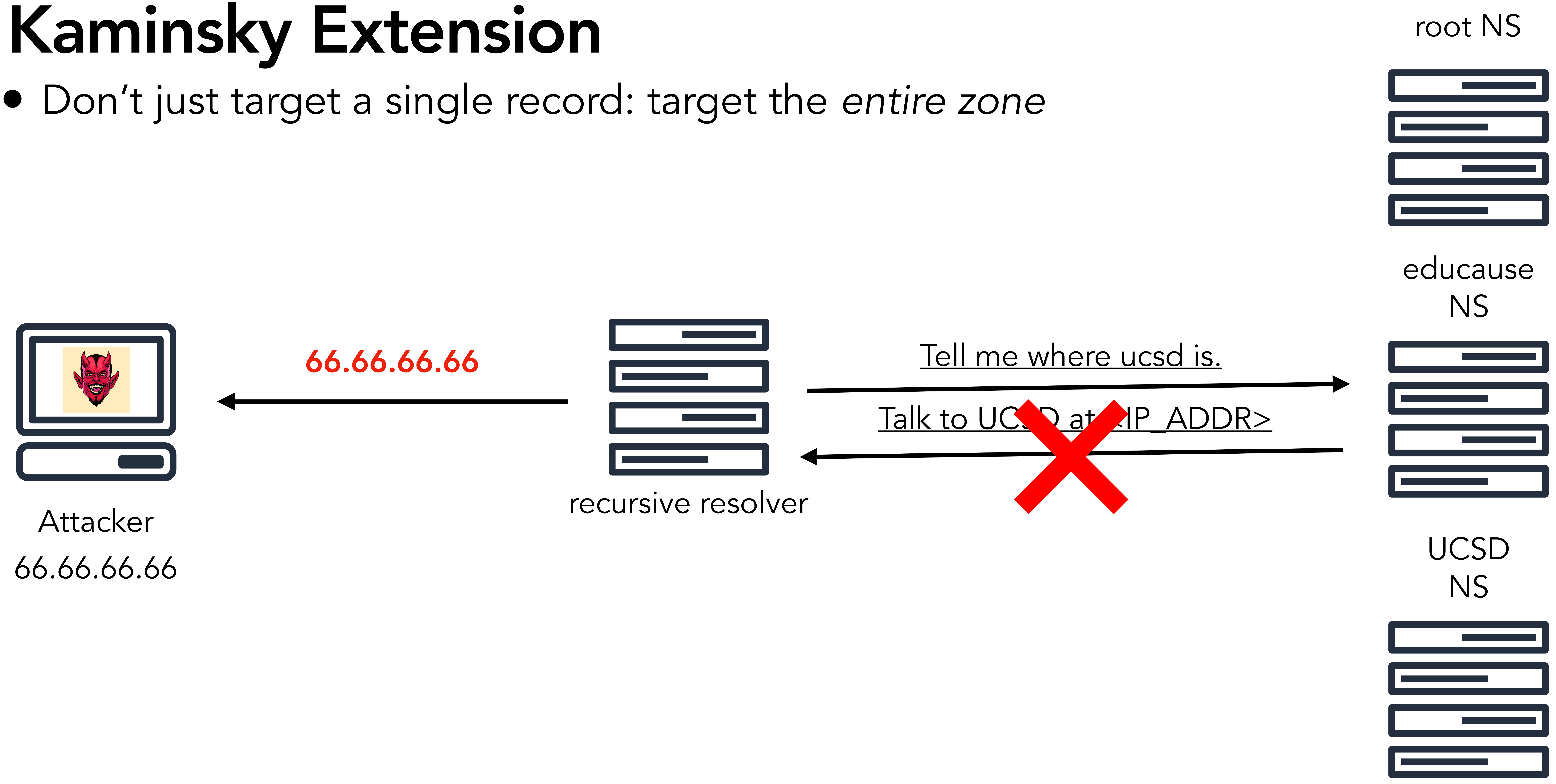
Kaminsky Extension

- Don’t just target a single record: target the *entire zone*



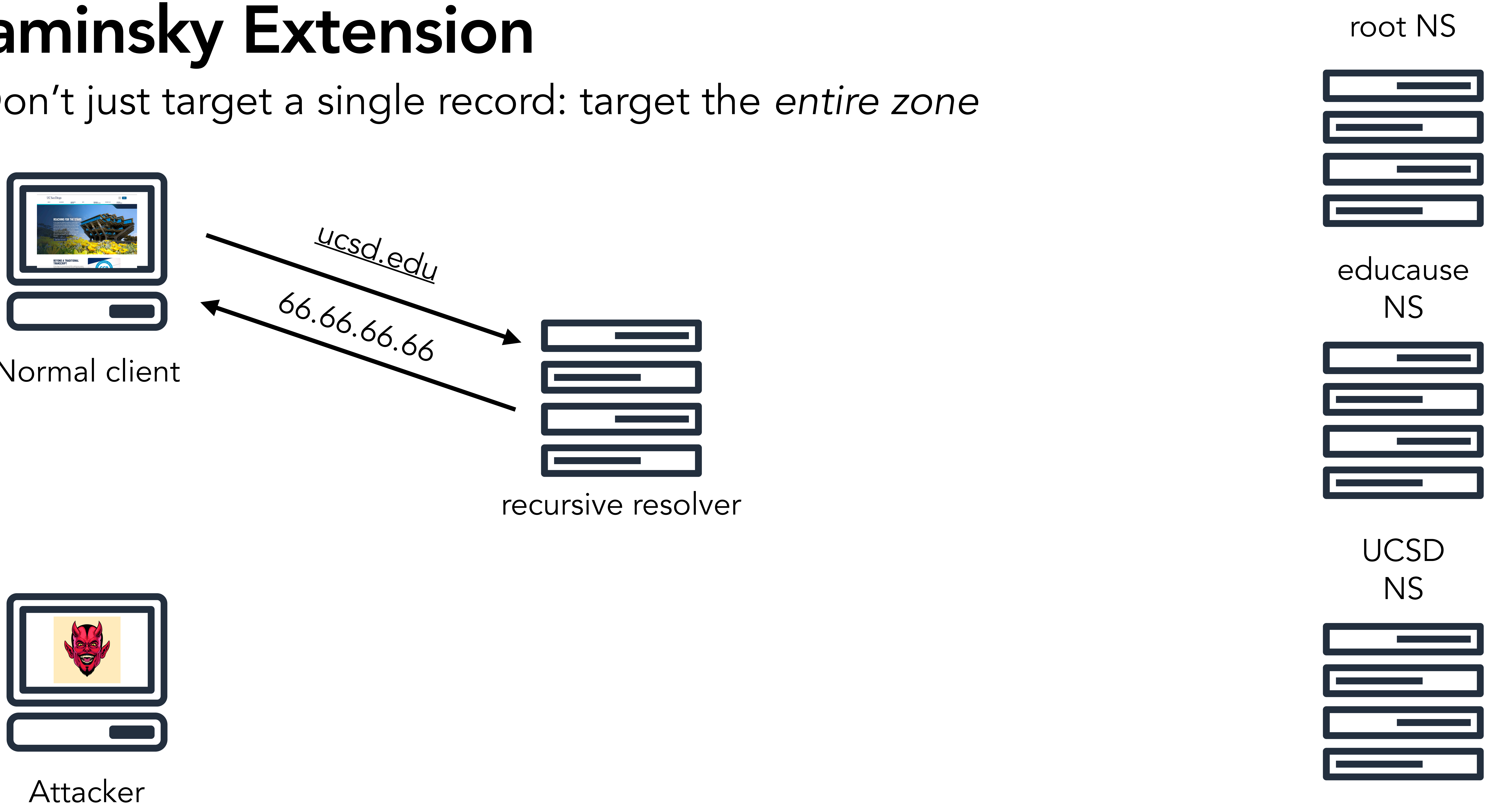
Kaminsky Extension

- Don't just target a single record: target the *entire zone*



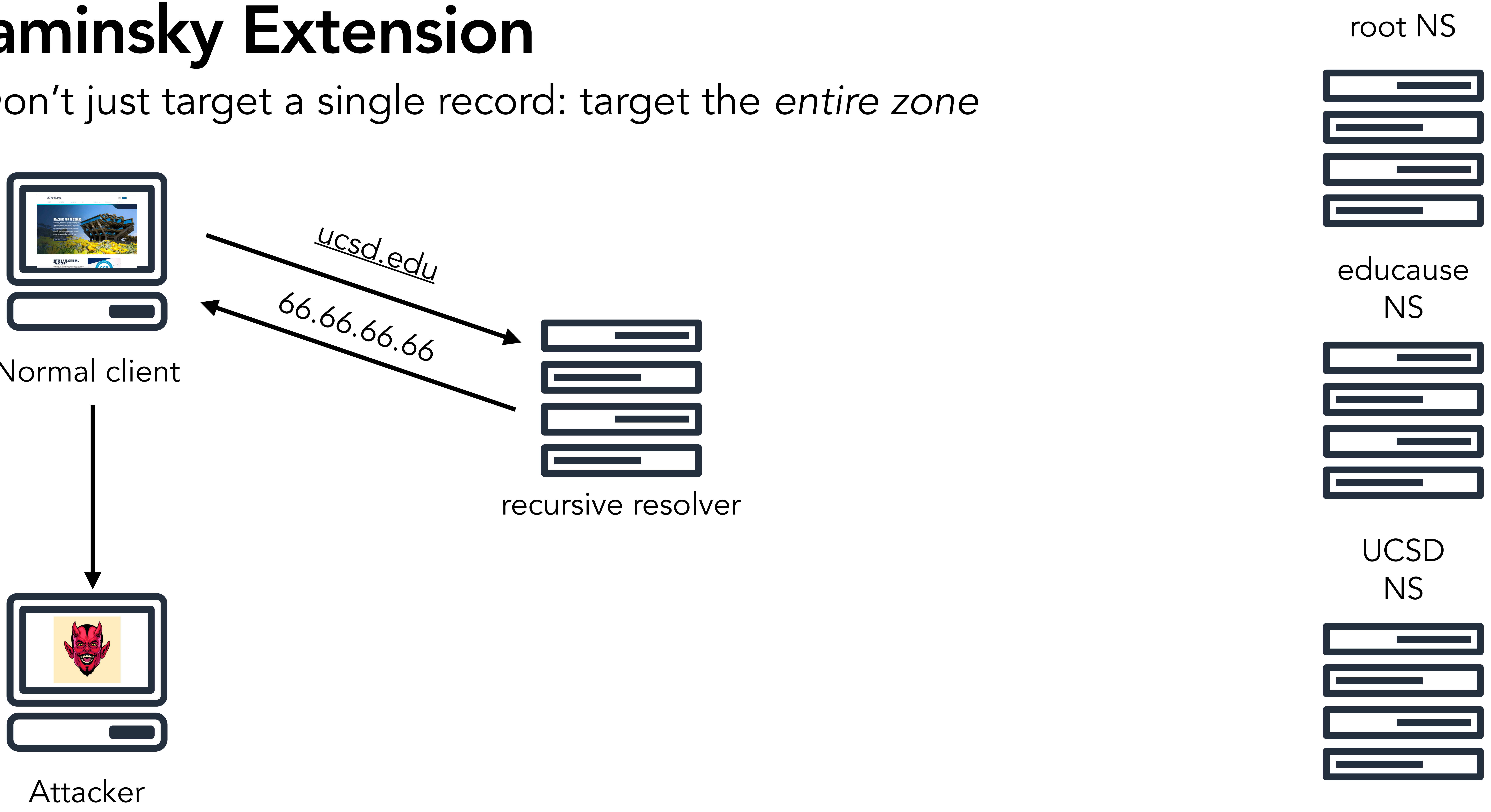
Kaminsky Extension

- Don’t just target a single record: target the *entire zone*



Kaminsky Extension

- Don’t just target a single record: target the *entire zone*



Assumptions

- What is the critical assumption baked into this attack?

Assumptions

- What is the critical assumption baked into this attack?
  - Target domain is not in the cache!

Assumptions

- What is the critical assumption baked into this attack?
    - Target domain is not in the cache!
- How do we get things out of the cache…?

Assumptions

- What is the critical assumption baked into this attack?
  - Target domain is not in the cache!
- How do we get things out of the cache…?
  - Flood the recursive with **lots** of queries

Countermeasures

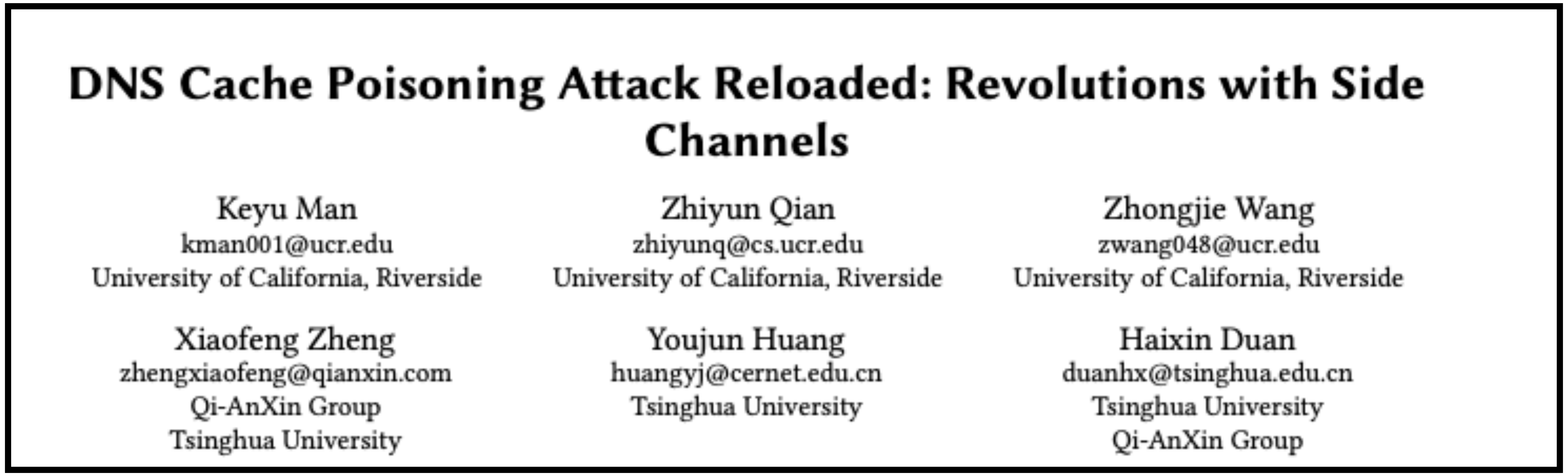
- What is the fundamental exploit in this attack?

Countermeasures

- What is the fundamental exploit in this attack?
    - QID leakage, and a **small** attack surface (16-bit query ID)
    - Ports were static across every request
- When QIDs are randomized and ports are randomized, attack becomes **very hard to execute**

Countermeasures

- What is the fundamental exploit in this attack?
    - QID leakage, and a **small** attack surface (16-bit query ID)
    - Ports were static across every request
- When QIDs are randomized and ports are randomized, attack becomes **very hard to execute**



DNS Cache Poisoning Attack Reloaded: Revolutions with Side Channels

Keyu Man
kman001@ucr.edu
University of California, Riverside

Zhiyun Qian
zhiyunq@cs.ucr.edu
University of California, Riverside

Zhongjie Wang
zwang048@ucr.edu
University of California, Riverside

Xiaofeng Zheng
zhengxiaofeng@qianxin.com
Qi-AnXin Group
Tsinghua University

Youjun Huang
huangyj@cernet.edu.cn
Tsinghua University

Haixin Duan
duanhx@tsinghua.edu.cn
Tsinghua University
Qi-AnXin Group

Discussion

- What surprised you about this attack? What did this attack make you think about trust?
- What surprised you about DNS if anything at all?
  - What do we do about it?
- For more, see: DNSSec (PKI for DNS to sign named records)

Next time…

- Final week on networks, focused this time on **censorship techniques** via the Network
- Midpoint check-in due **tomorrow!**