The Web, Revisited

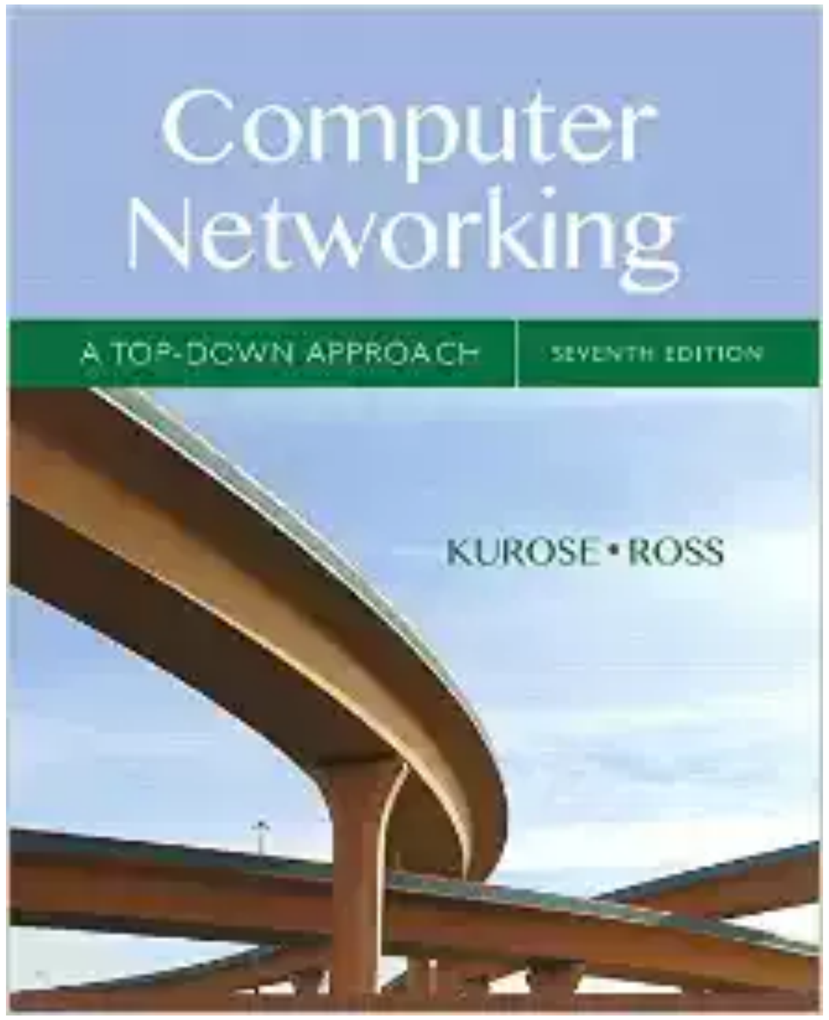
Mark Allman
mallman@case.edu

EECS 325/425
Fall 2018

"The black and white they cruise by …
… and watch us from the corner of their eye …"

Securing Web Content

Reading Along ...



- Web security, chapter 8

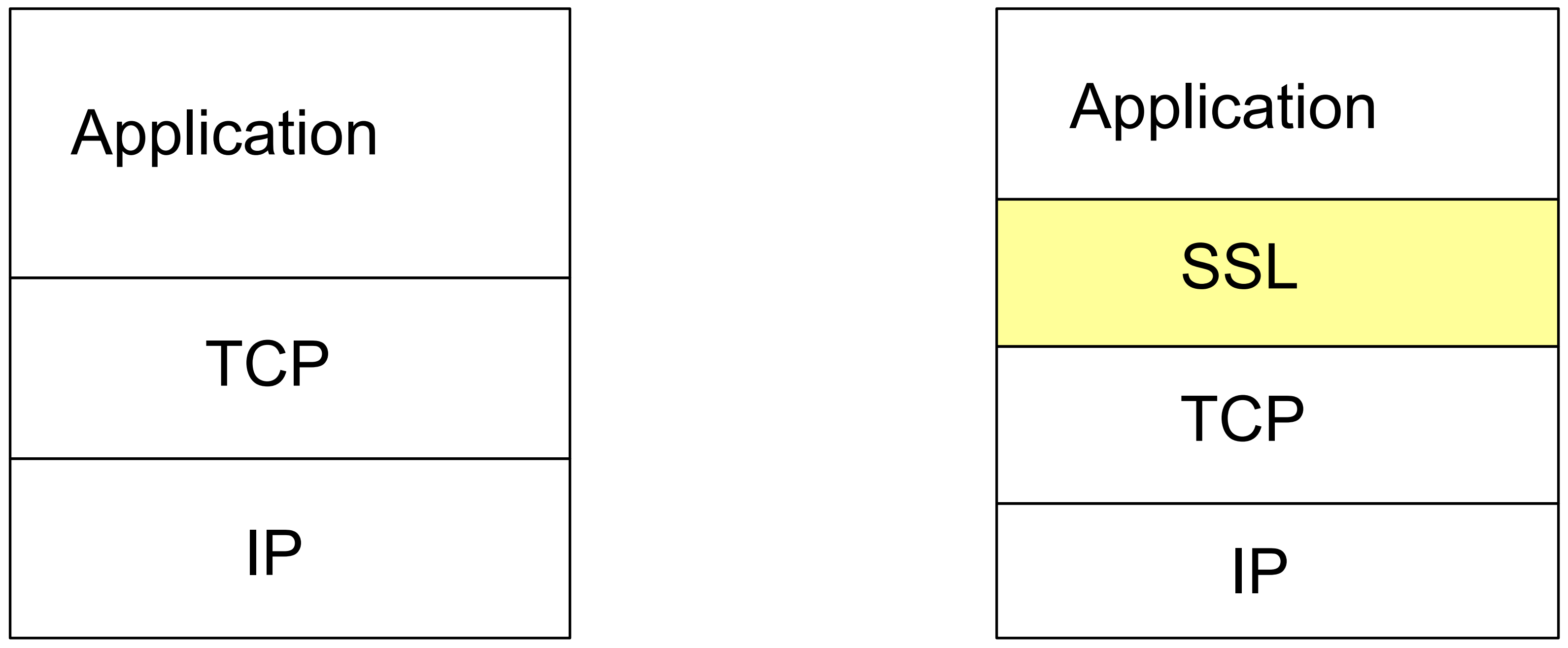
SSL: Secure Sockets Layer

- widely deployed security protocol
  - supported by almost all browsers, web servers
  - https
  - billions $/year over SSL
- mechanisms: [Woo 1994], implementation: Netscape
- variation -TLS: transport layer security, RFC 2246
- provides
  - *confidentiality*
  - *integrity*
  - *authentication*
- original goals:
  - Web e-commerce transactions
  - encryption (especially credit-card numbers)
  - Web-server authentication
  - optional client authentication
  - minimum hassle in doing business with new merchant
- available to all TCP applications
  - secure socket interface

SSL vs. TLS

- Secure Sockets Layer (SSL) ≈ Transport Layer Security (TLS)

SSL and TCP/IP



normal application

application with SSL

- SSL provides application programming interface (API) to applications
- C and Java SSL libraries/classes readily available

Toy SSL: a simple secure channel

Toy SSL: a simple secure channel

- *handshake:* Alice and Bob use their certificates, private keys to authenticate each other and exchange shared secret

Toy SSL: a simple secure channel

- *handshake:* Alice and Bob use their certificates, private keys to authenticate each other and exchange shared secret
- *key derivation:* Alice and Bob use shared secret to derive set of keys to secure the *data*

Toy SSL: a simple secure channel

- *handshake:* Alice and Bob use their certificates, private keys to authenticate each other and exchange shared secret
- *key derivation:* Alice and Bob use shared secret to derive set of keys to secure the *data*
- *data transfer:* data to be transferred is broken up into series of records

Toy SSL: a simple secure channel

- *handshake:* Alice and Bob use their certificates, private keys to authenticate each other and exchange shared secret
- *key derivation:* Alice and Bob use shared secret to derive set of keys to secure the *data*
- *data transfer:* data to be transferred is broken up into series of records
- *connection closure:* special messages to securely close connection

Toy: data records

Toy: data records

- why not encrypt data in constant stream as we write it to TCP?

Toy: data records

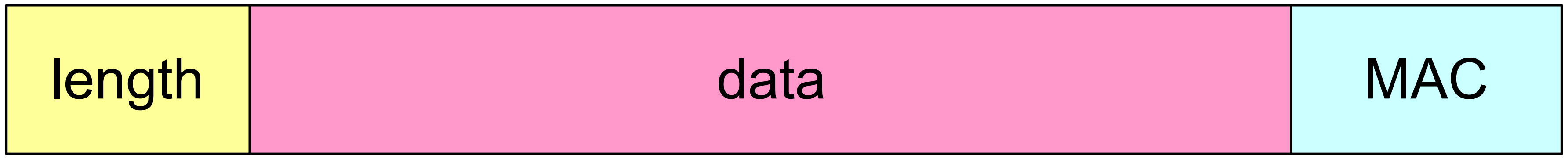
- why not encrypt data in constant stream as we write it to TCP?
  - when would be check integrity? If at end, no message integrity until all data processed.
  - e.g., with instant messaging, how can we do integrity check over all bytes sent before displaying?

Toy: data records

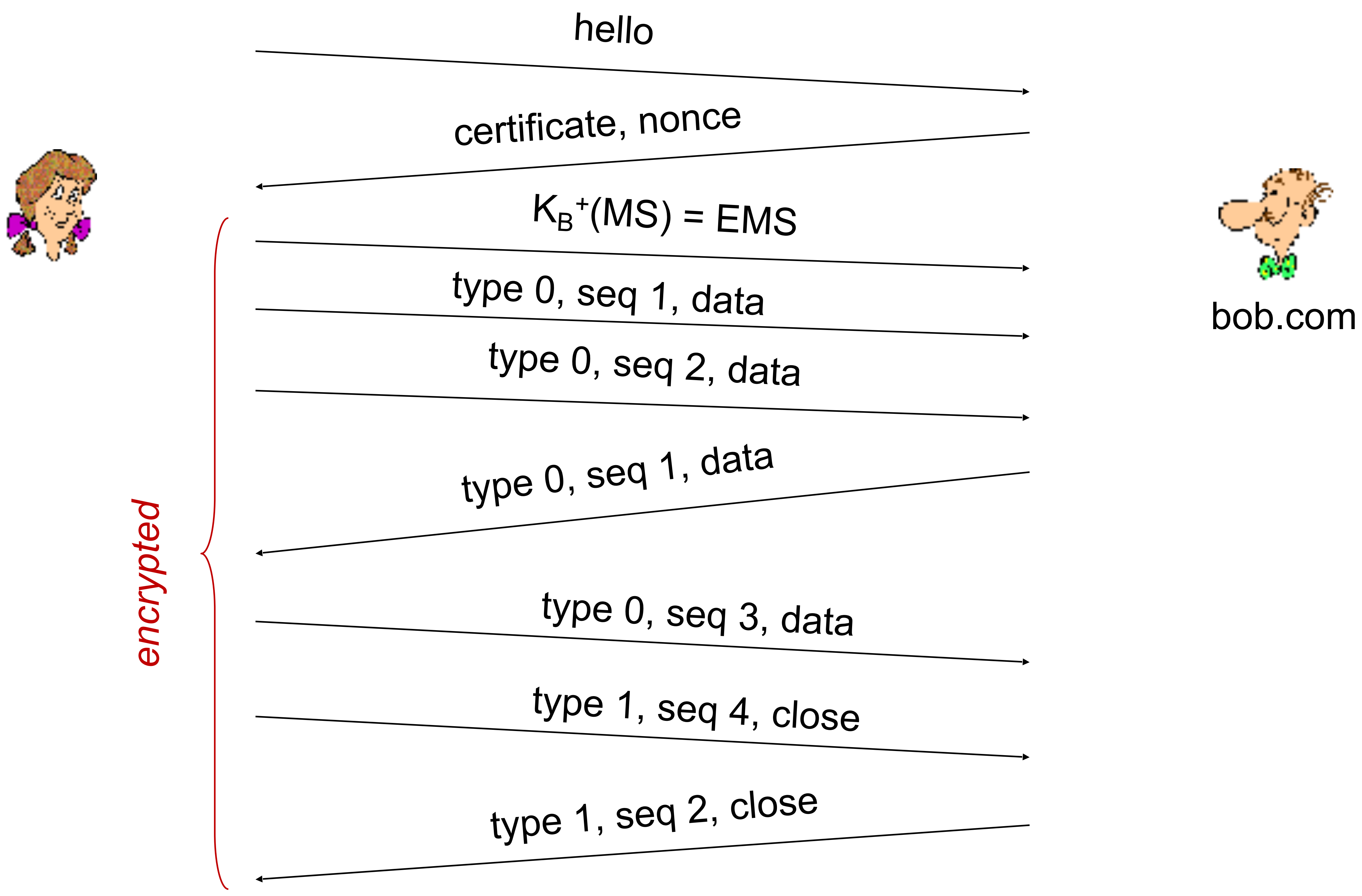
- why not encrypt data in constant stream as we write it to TCP?
  - when would be check integrity? If at end, no message integrity until all data processed.
  - e.g., with instant messaging, how can we do integrity check over all bytes sent before displaying?
- instead, break stream in series of records
  - each record carries a MAC
  - receiver can act on each record as it arrives

Toy: data records

- why not encrypt data in constant stream as we write it to TCP?
  - when would be check integrity? If at end, no message integrity until all data processed.
  - e.g., with instant messaging, how can we do integrity check over all bytes sent before displaying?
- instead, break stream in series of records
  - each record carries a MAC
  - receiver can act on each record as it arrives
- issue: in record, receiver needs to distinguish MAC from data
  - want to use variable-length records



Toy SSL: summary



Toy SSL isn't complete

- how long are fields?
- which encryption protocols?
- want negotiation?
  - allow client and server to support different encryption algorithms
  - allow client and server to choose together specific algorithm before data transfer

A New Web “Transport” Protocol

HTTP-over-TCP

- "The web" is a combination of several pieces of technology that are used together
    - but, pieced together
    - not from an integrated design
    - shows the power of generality …
      … but also the limitations

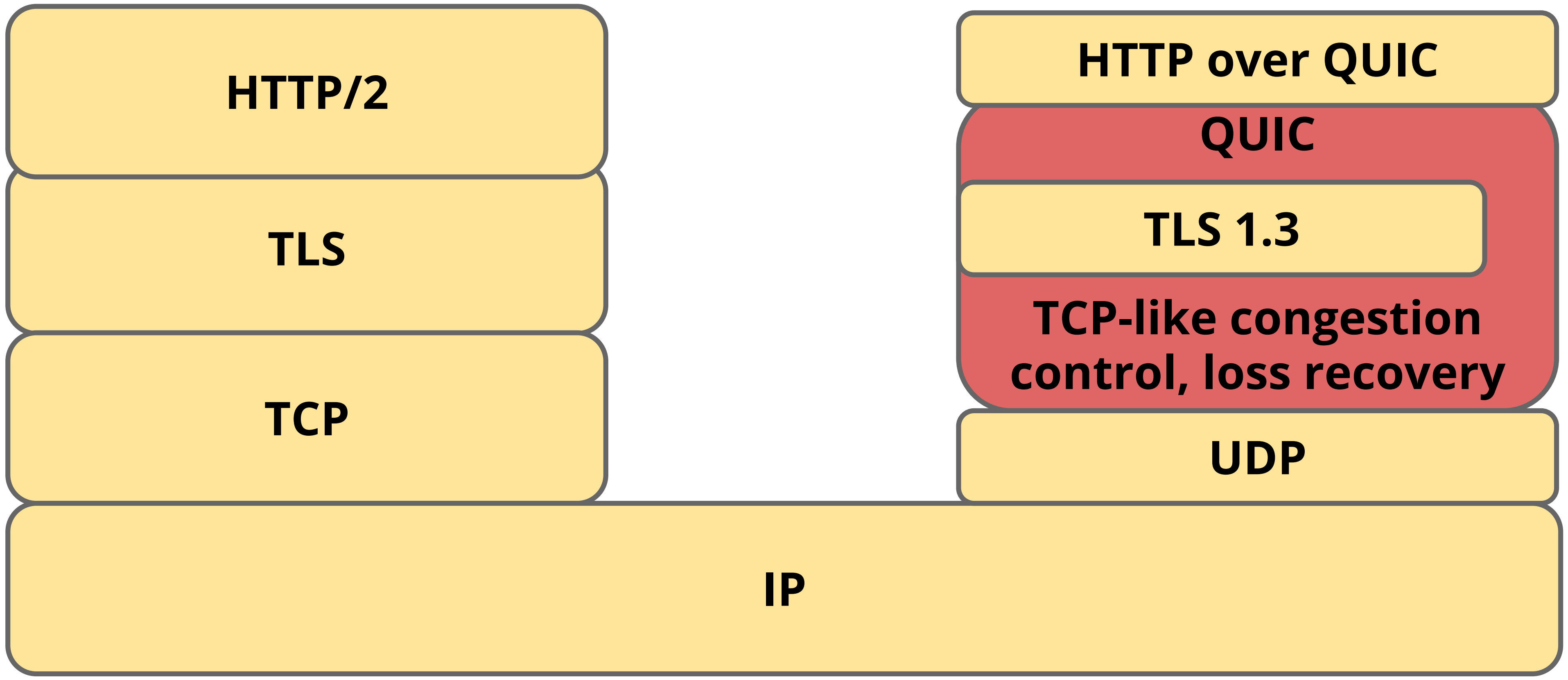
HTTP-over-TCP

- Web transactions
  - 1 RTT to setup TCP connection
  - 2 RTTs to setup TLS
  - Then, HTTP request / response

QUIC

- Developed by Google in 2014
- Goal: improve web load latency, video playback experience
- Initially deployed between Chrome & Google services
  - One-third of Google traffic now using QUIC
- Now, being standardized and used by others

QUIC



QUIC Tutorial, IETF-98

HTTP-over-QUIC

- Web transactions
  - 0 RTTs to setup connection to known server (common)
  - 1 RTT if crypto keys are not new
  - 2 RTTs if QUIC version negotiation is required
  - Then, HTTP request / response

QUIC

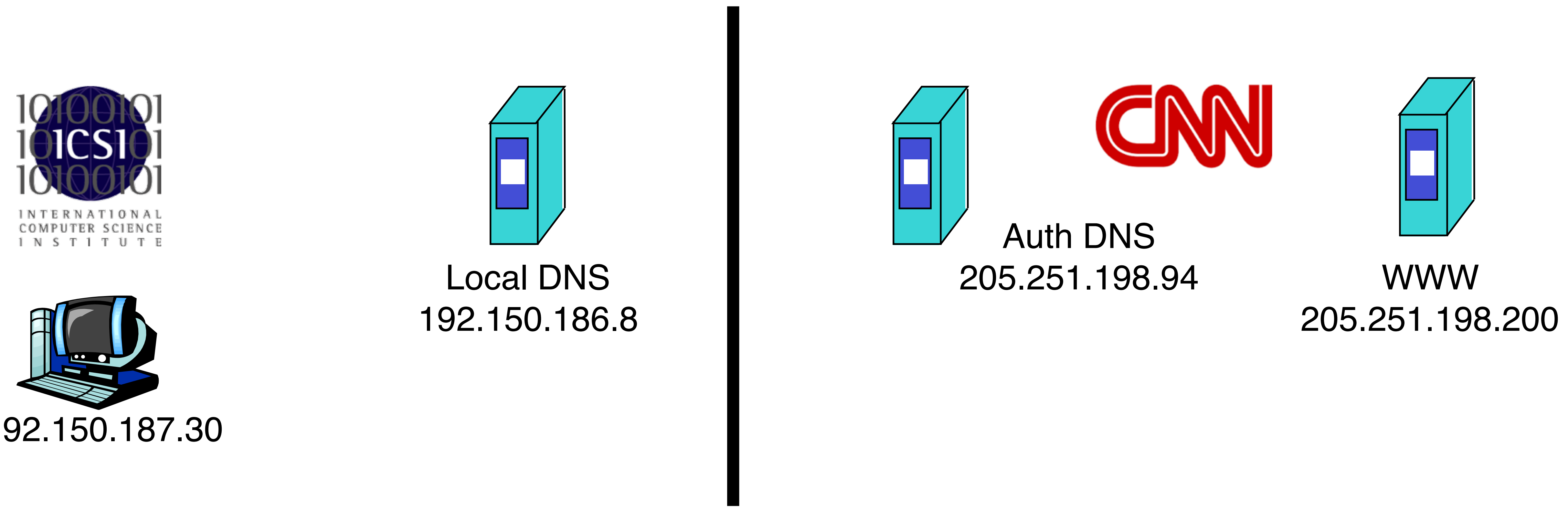
- QUIC borrows liberally from many different technologies
  - deployment eased by riding on top of UDP
  - uses TLS
  - congestion control is based on decades of work with TCP CC

QUIC

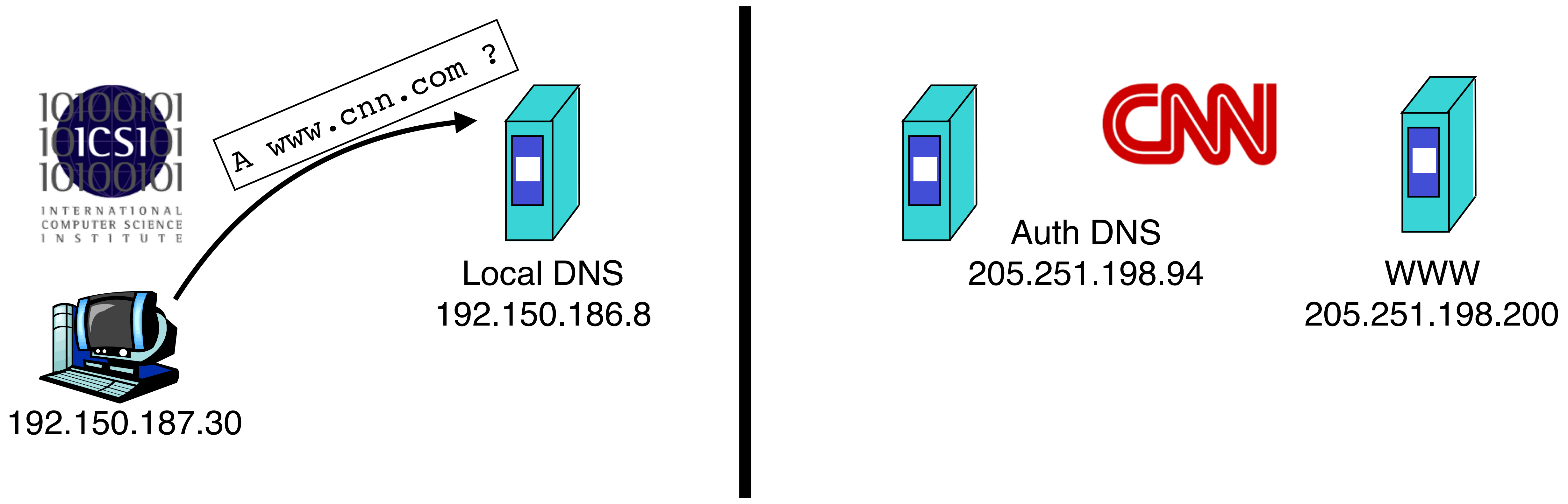
- signaling is richer, but based on experience
  - e.g., TCP can send 3 SACK blocks …
    … but QUIC can send 256
  - e.g., built more machinery to avoid RTO

Content Distribution Networks

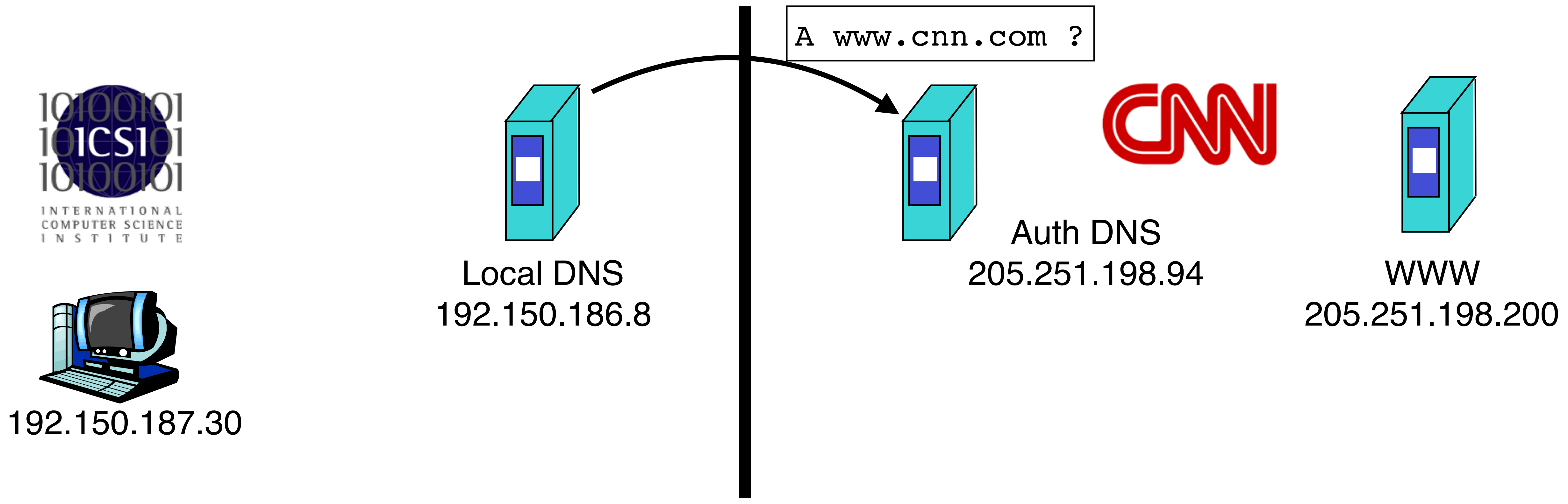
Web Transactions



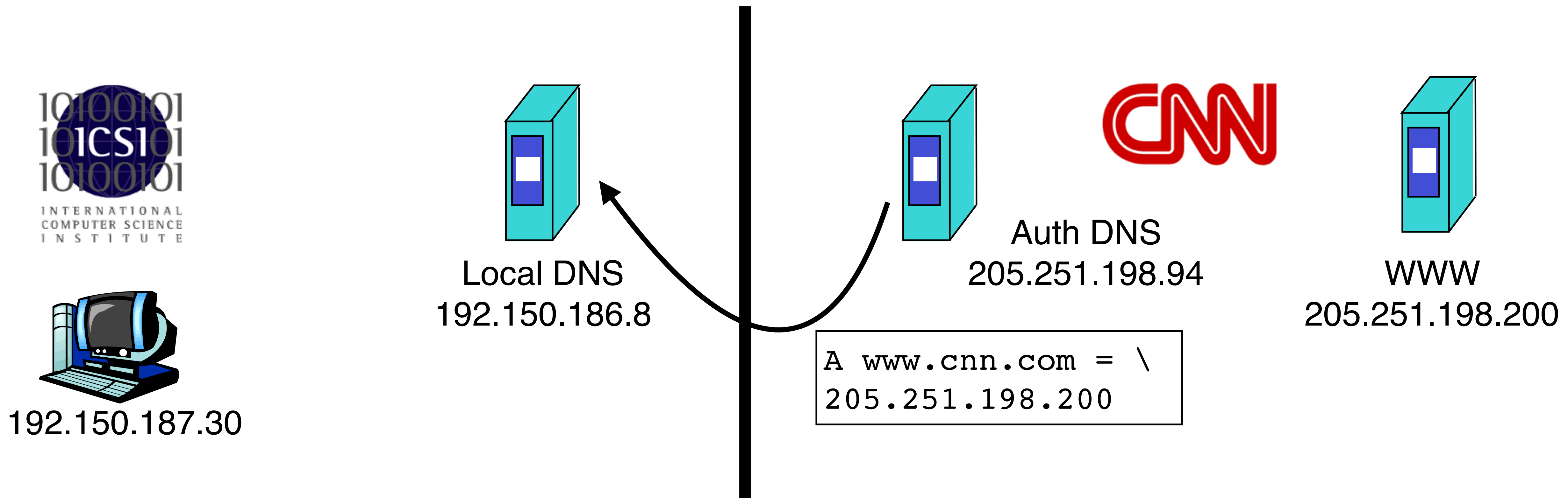
Web Transactions



Web Transactions

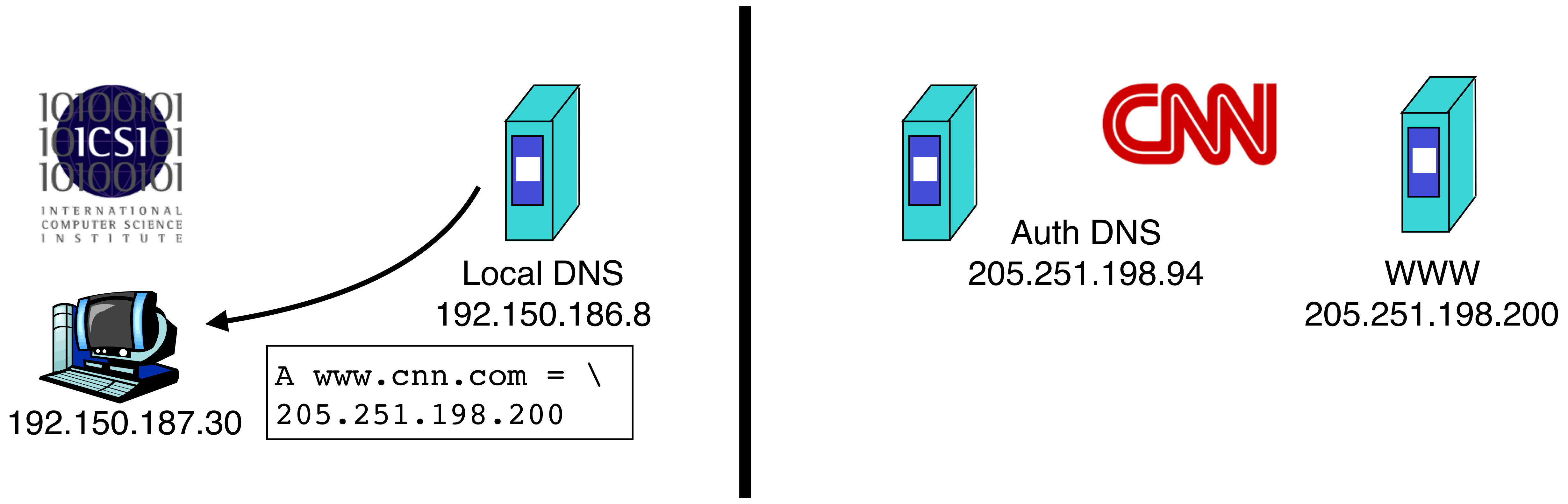


Web Transactions



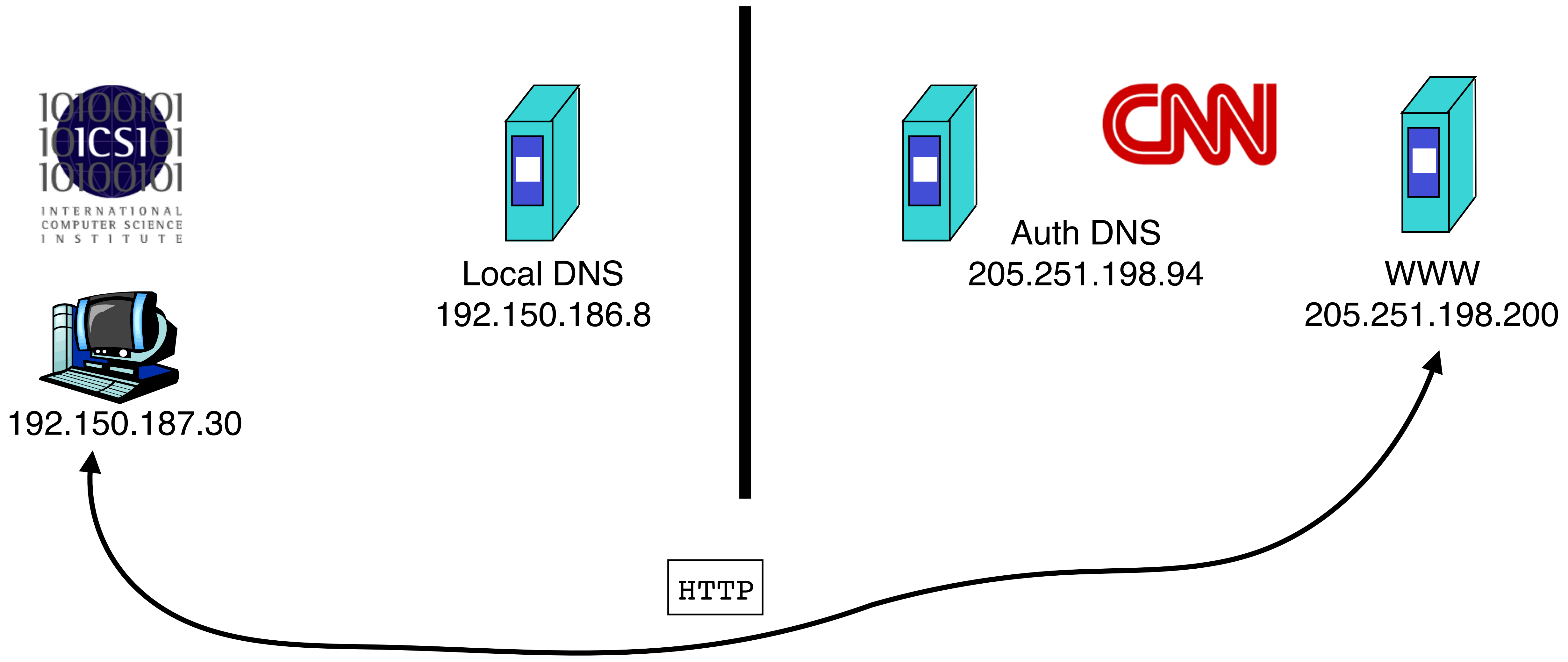
```
A www.cnn.com = \
205.251.198.200
```

Web Transactions

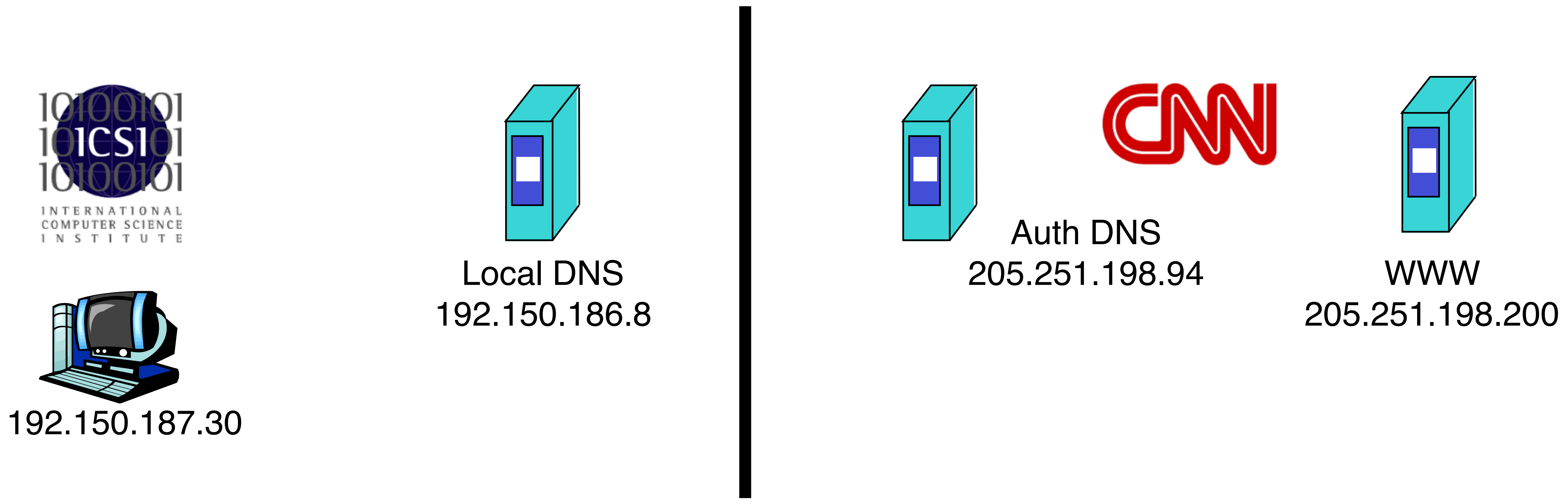


```
A www.cnn.com = \
205.251.198.200
```

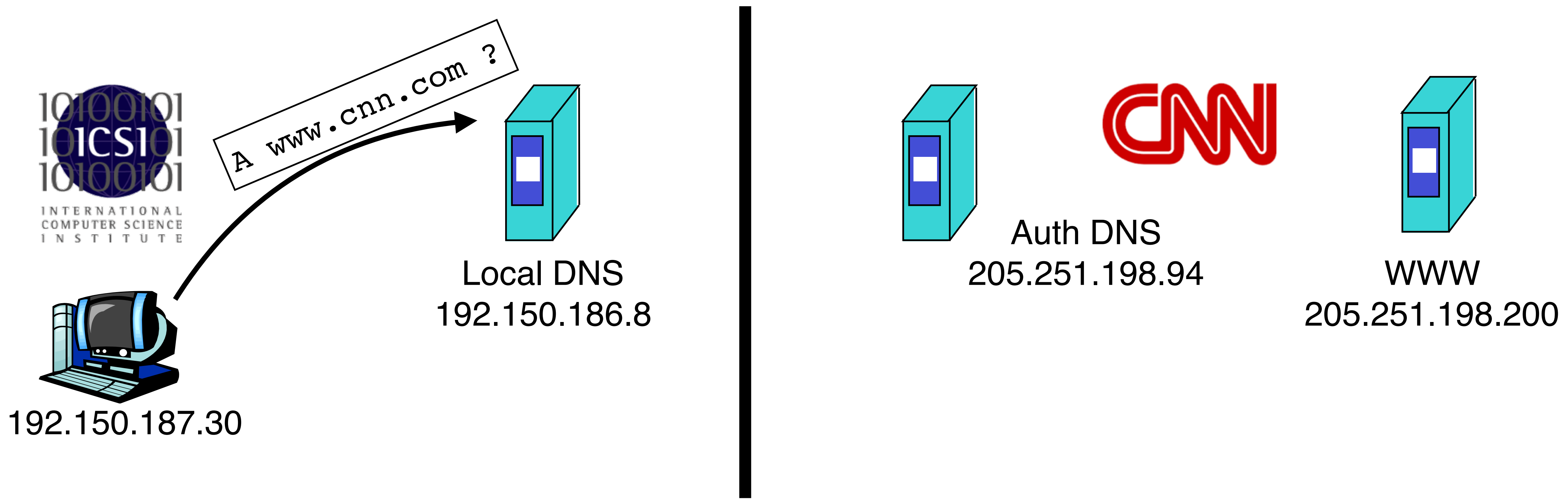
Web Transactions



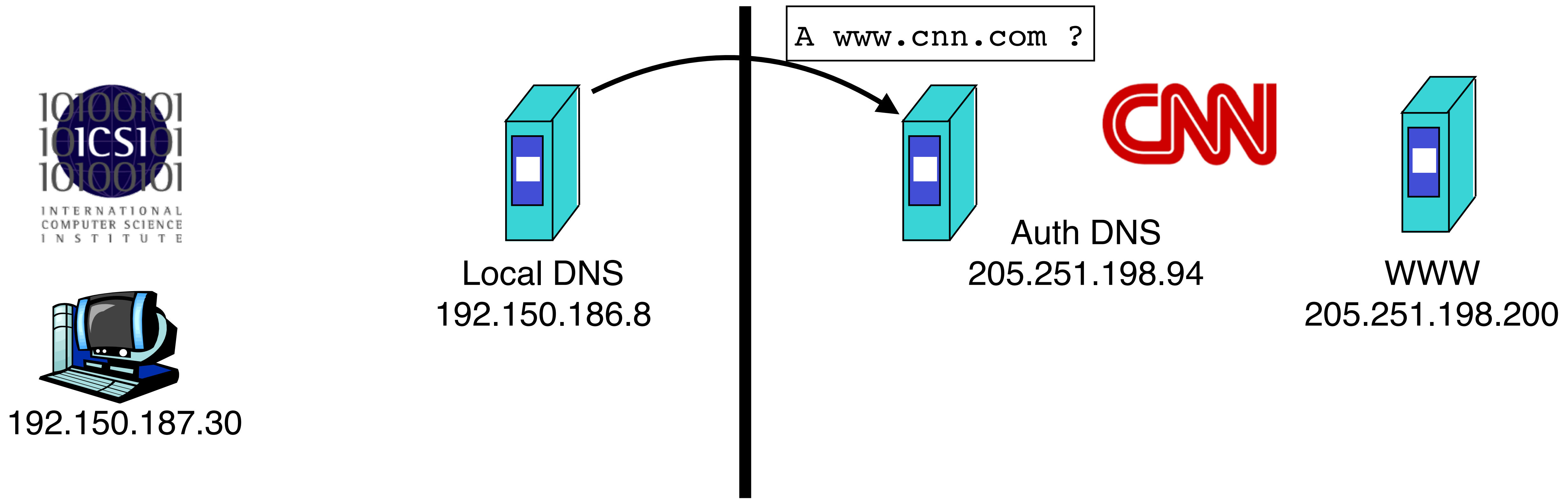
Web Trans. with CDNs



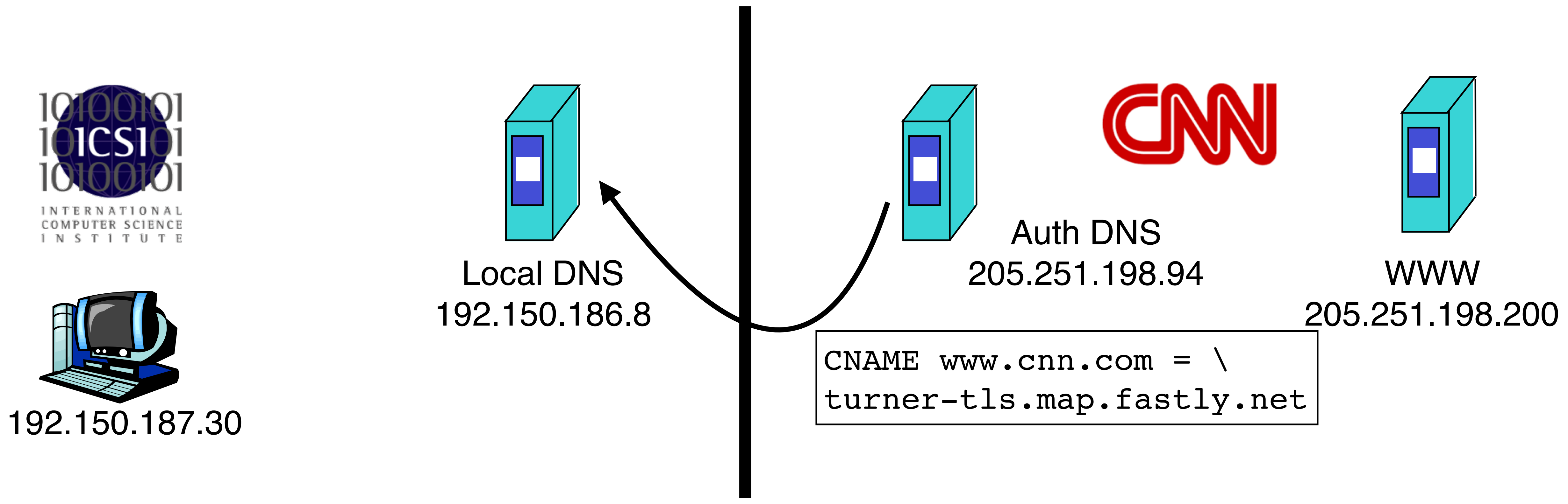
Web Trans. with CDNs



Web Trans. with CDNs

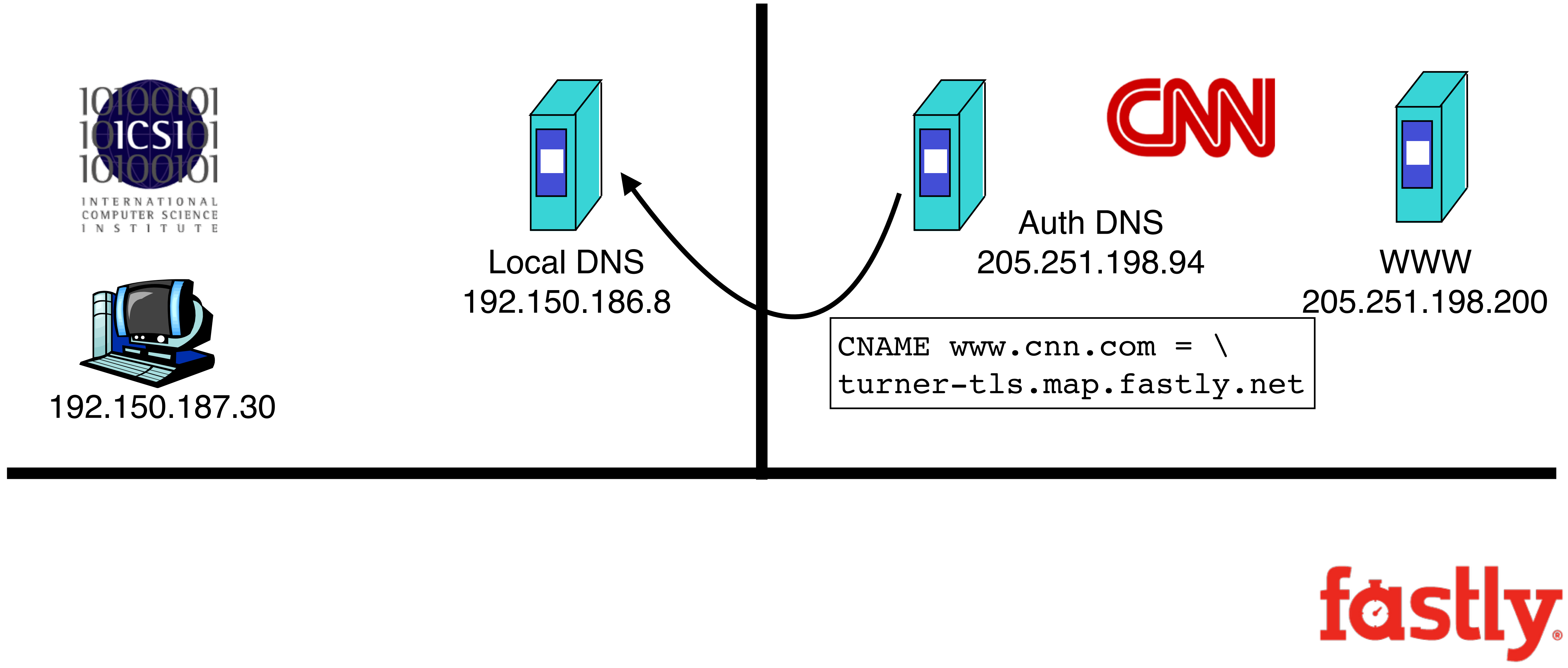


Web Trans. with CDNs



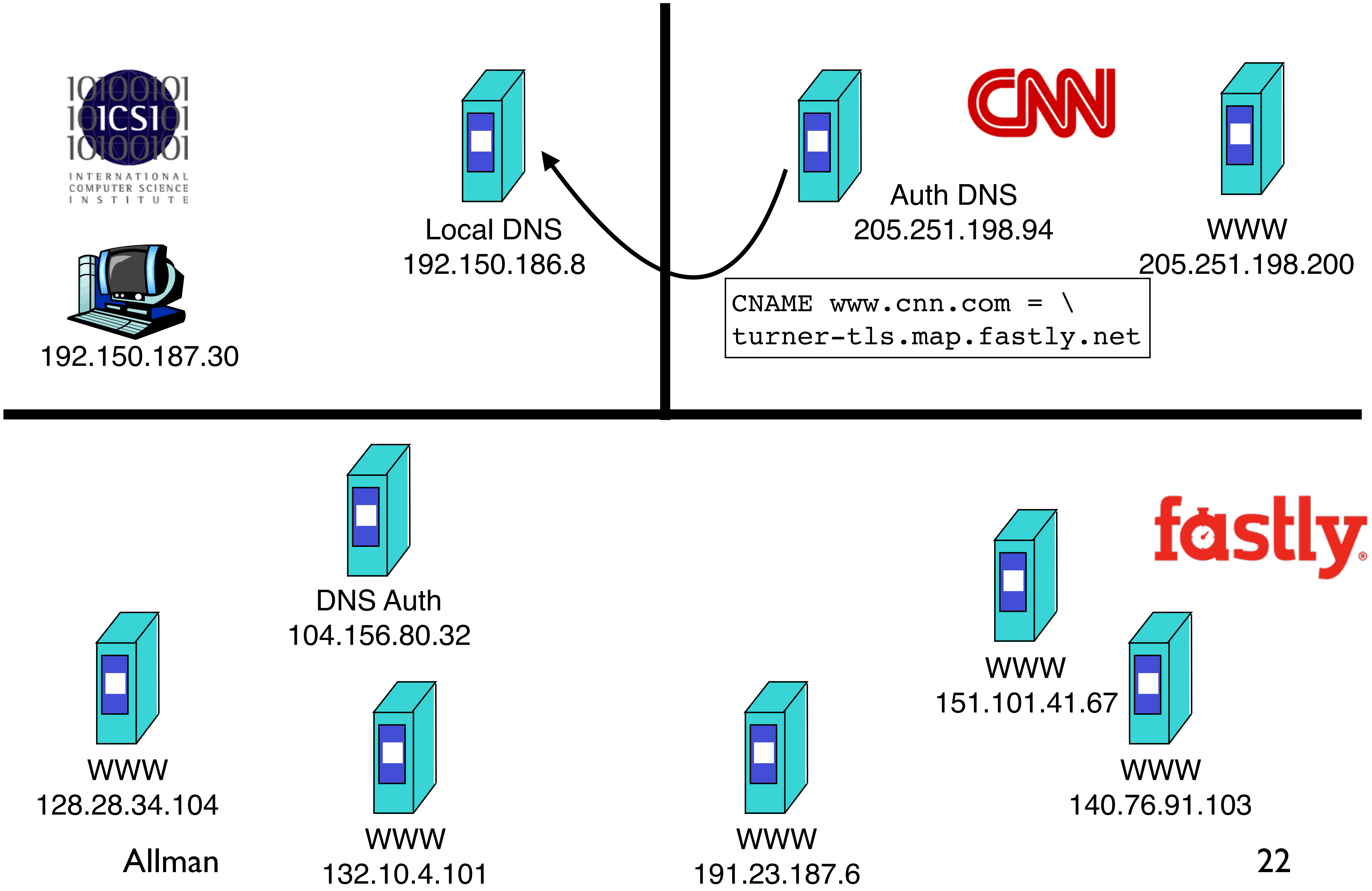
```
CNAME www.cnn.com = \
turner-tls.map.fastly.net
```

Web Trans. with CDNs



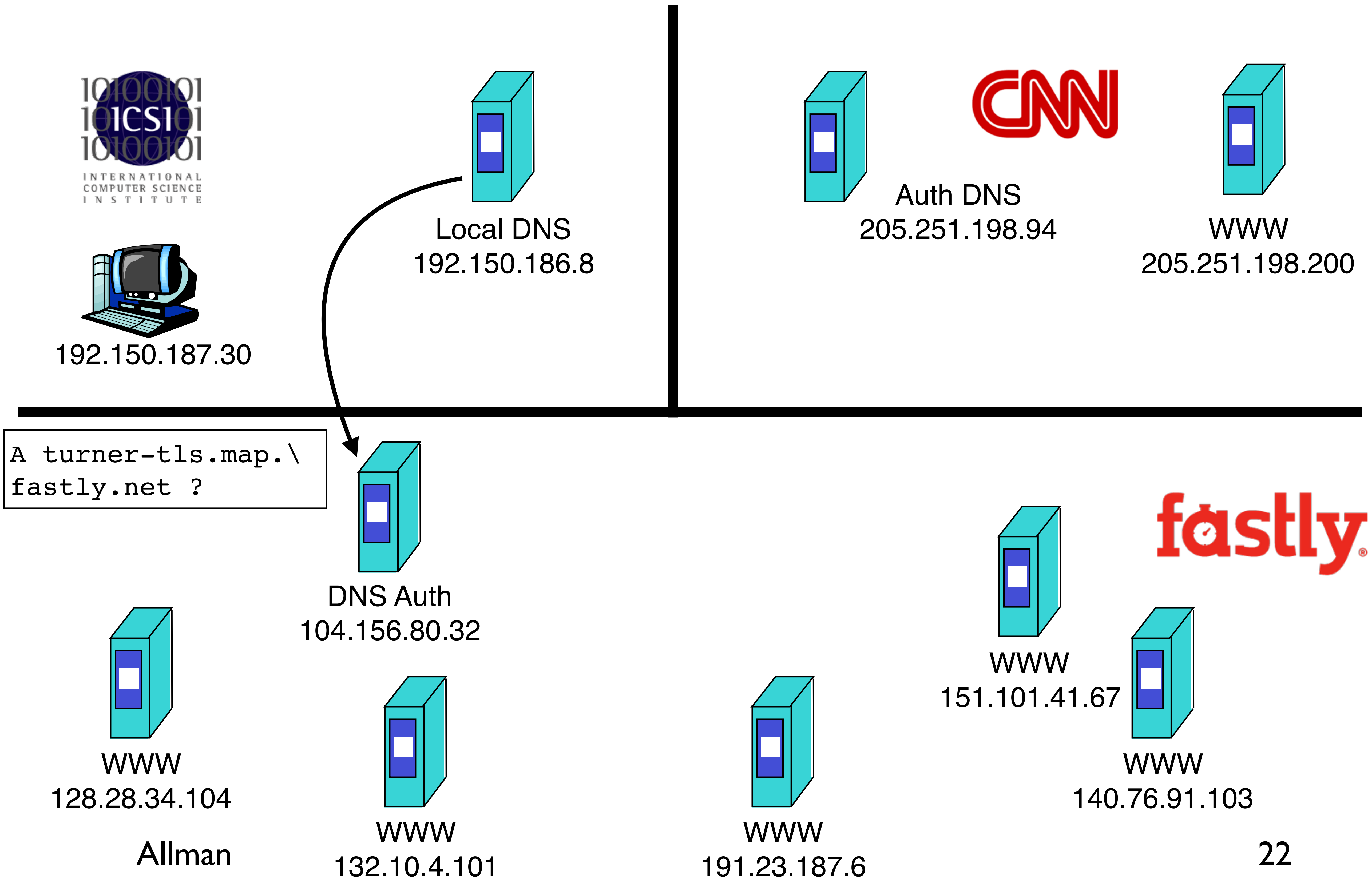
```
CNAME www.cnn.com = \
turner-tls.map.fastly.net
```

Web Trans. with CDNs



Allman

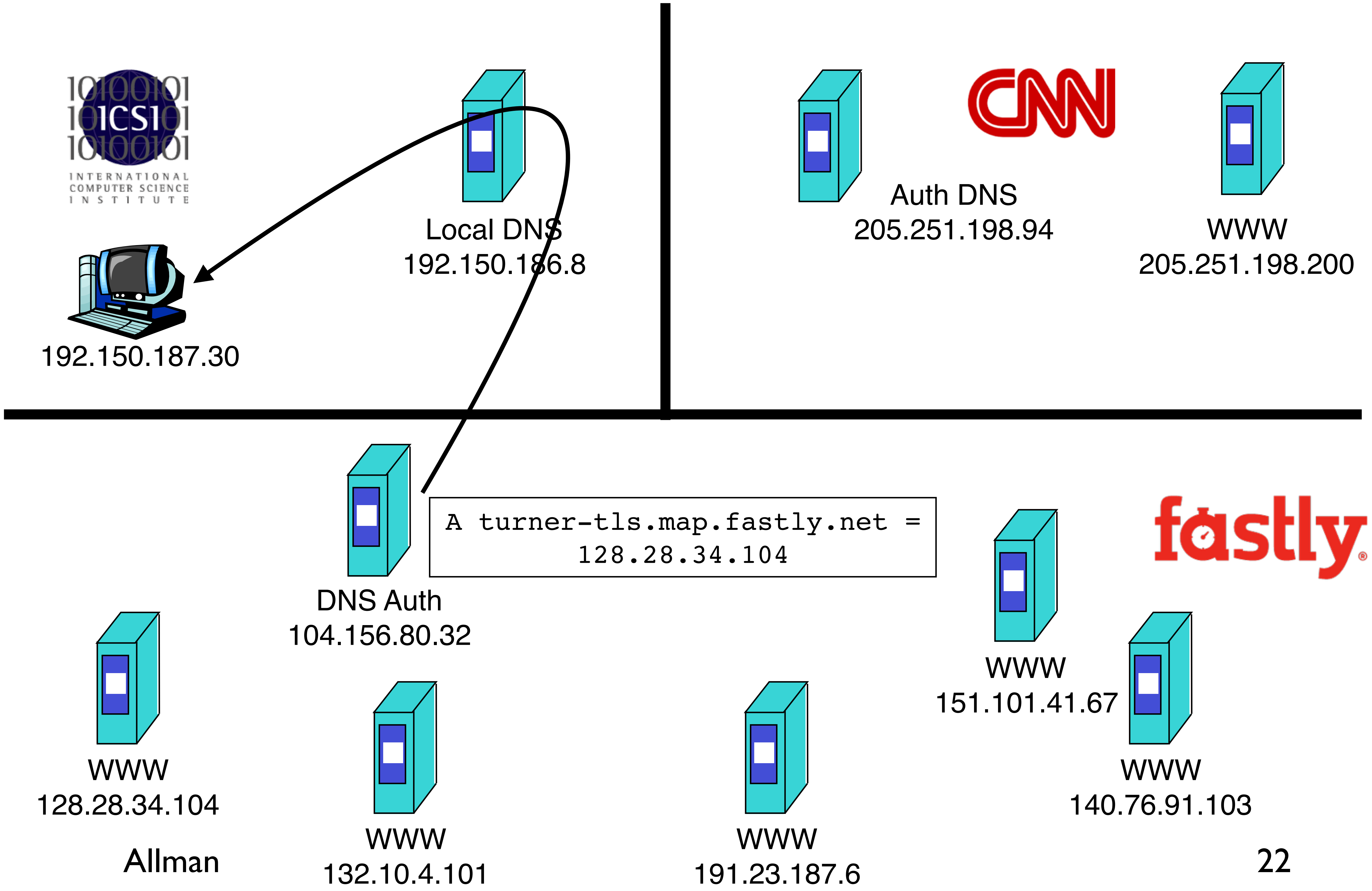
Web Trans. with CDNs



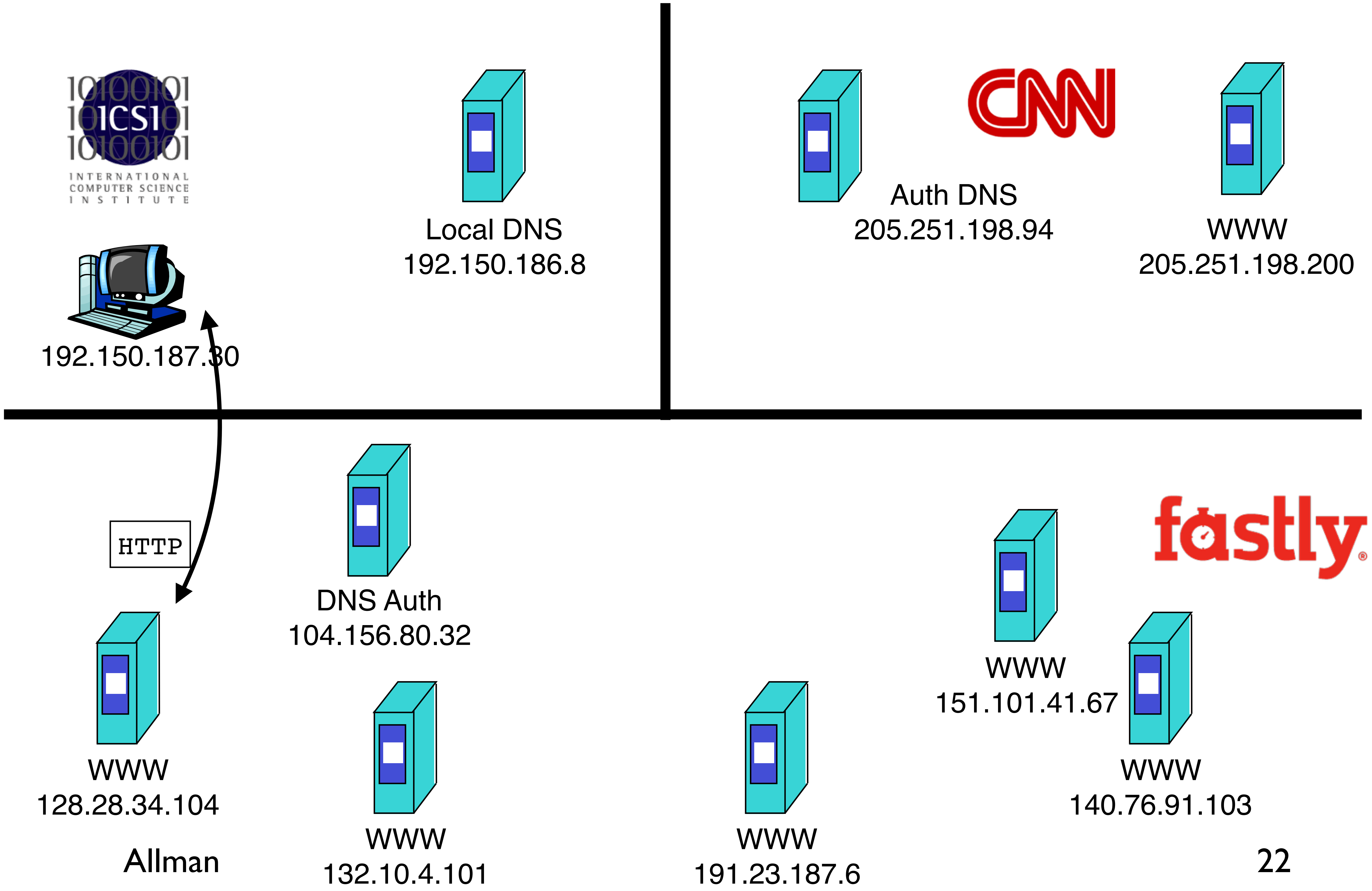
```
A turner-tls.map.\
fastly.net ?
```

Allman

Web Trans. with CDNs

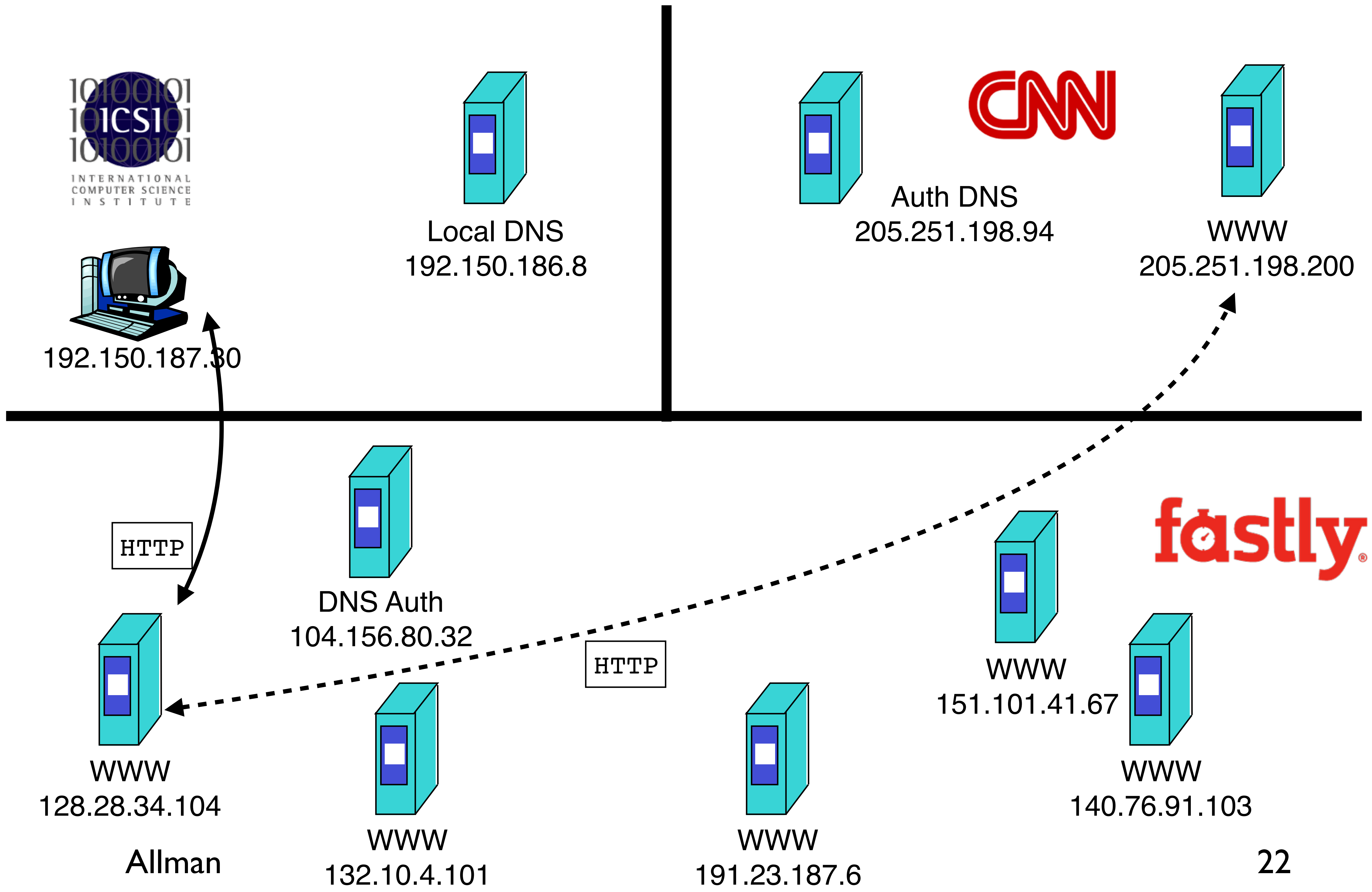


Web Trans. with CDNs



Allman

Web Trans. with CDNs



Allman

CDNs

CDNs

- Why arrange things in this fashion?

CDNs

- Why arrange things in this fashion?
- Advantages
  - sheds load from content providers
  - helps mitigate DDoS
  - perhaps moves data closer to the user
    - so, provides quicker retrieval
    - better “quality of experience” (QoE)

CDNs

CDNs

- Disadvantages

CDNs

- Disadvantages
  - content provider loses fine-grain control

CDNs

- Disadvantages
    - content provider loses fine-grain control
    - content provider loses visibility

CDNs

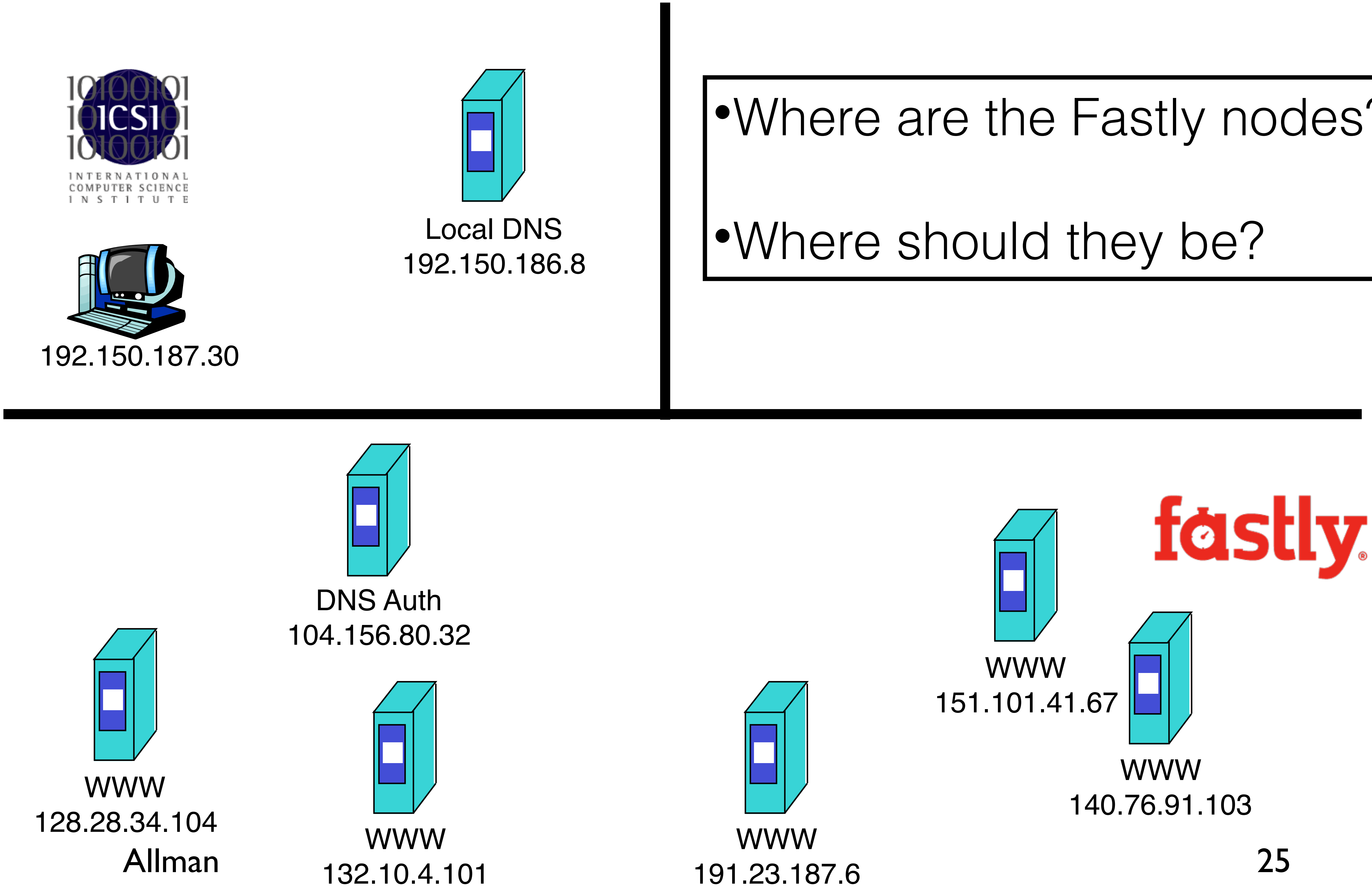
- Disadvantages
  - content provider loses fine-grain control
  - content provider loses visibility
    - e.g., for accounting

CDNs

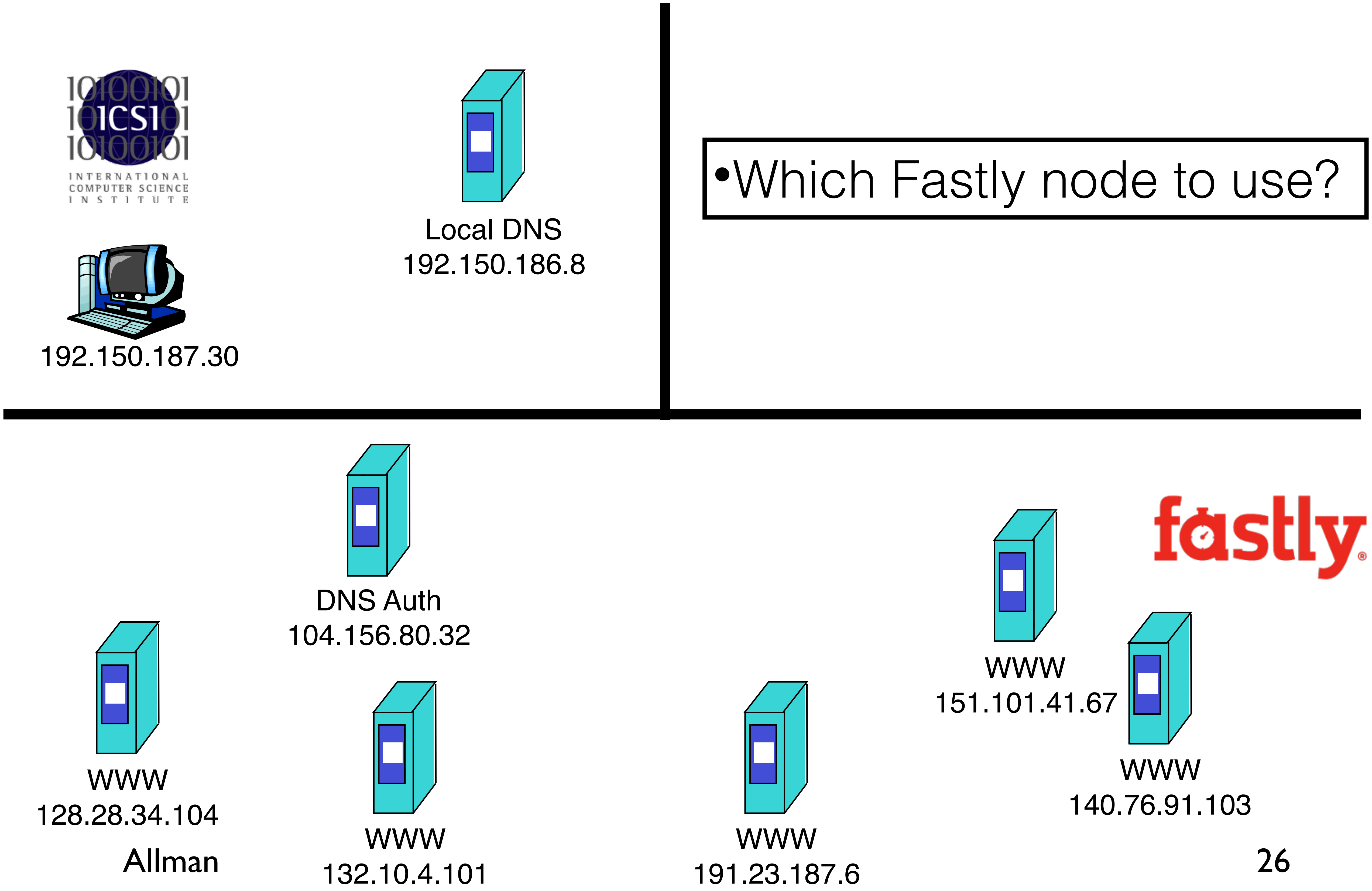
- Disadvantages
  - content provider loses fine-grain control
  - content provider loses visibility
    - e.g., for accounting
  - content provider must trust CDN

CDNs

- Where are the Fastly nodes?
- Where should they be?



CDNs



- Which Fastly node to use?

Allman

CDNs

CDNs

- How should we set the DNS TTL?

CDNs

- How should we set the DNS TTL?
- Some CDNs use anycast routing
  - why?

CDNs

- Big distributed systems with lots of tradeoffs