

DNS Security

- The Domain Name Service (DNS) translates human-readable names to IP addresses
 - E.g., thesiger.cs.ucla.edu translates to 131.179.192.144
 - DNS also provides other similar services
- It wasn't designed with security in mind

DNS Threats

- Threats to name lookup secrecy
 - Definition of DNS system says this data isn't secret
- Threats to DNS information integrity
 - Very important, since everything trusts that this translation is correct
- Threats to DNS availability
 - Potential to disrupt Internet service

What Could Really Go Wrong?

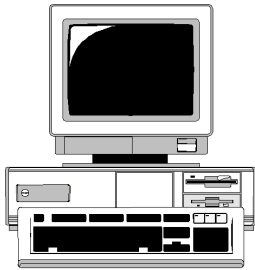
- DNS lookups could be faked
 - Meaning packets go to the wrong place
- The DNS service could be subject to a DoS attack
 - Or could be used to amplify one
- Attackers could “bug” a DNS server to learn what users are looking up

Where Does the Threat Occur?

- Unlike routing, threat can occur in several places
 - At DNS servers
 - But also at DNS clients
 - Which is almost everyone
- Core problem is that DNS responses aren't authenticated

The DNS Lookup Process

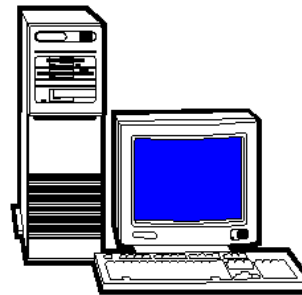
```
lookup thesiger.cs.ucla.edu
```



```
ping thesiger.cs.ucla.edu
```

Should result in a ping
packet being sent to
131.179.191.144

```
answer 131.179.191.144
```



If the answer is
wrong, in standard
DNS the client is
screwed

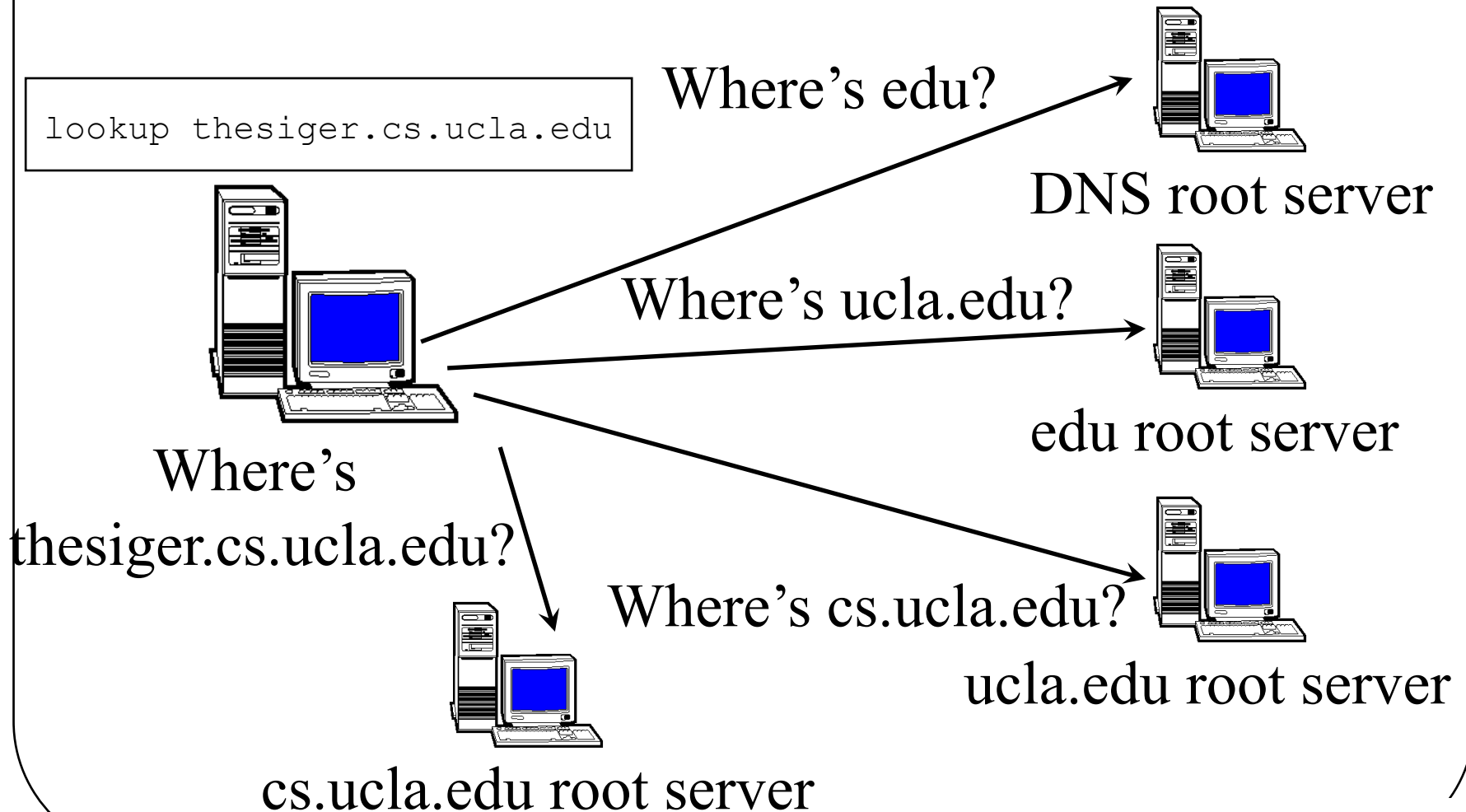
How Did the DNS Server Perform the Lookup?

- Leaving aside details, it has a table of translations between names and addresses
- It looked up thesiger.cs.ucla.edu in the table
- And replied with whatever the address was

Where Did That Table Come From?

- Ultimately, the table entries are created by those owning the domains
 - On a good day . . .
- And stored at servers that are authoritative for that domain
- In this case, the UCLA Computer Science Department DNS server ultimately stored it
- Other servers use a hierarchical lookup method to find the translation when needed

Doing Hierarchical Translation



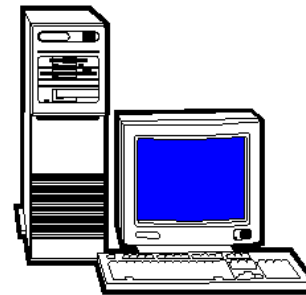
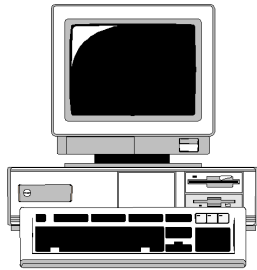
Where Can This Go Wrong?

- Someone can spoof the answer from a DNS server
 - Relatively easy, since UDP is used
- One of the DNS servers can lie
- Someone can corrupt the database of one of the DNS servers

The Spoofing Problem

```
lookup thesiger.cs.ucla.edu
```

```
answer 131.179.191.144
```



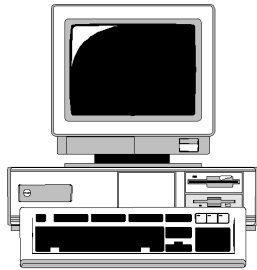
Unfortunately,
most DNS stub
resolvers will
take the first
answer

```
answer 97.22.101.53
```

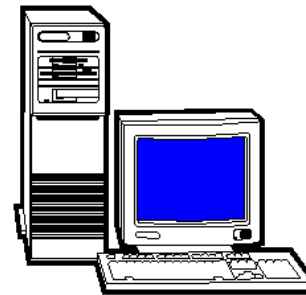


DNS Servers Lying

```
lookup thesiger.cs.ucla.edu
```



```
answer 97.22.101.53
```

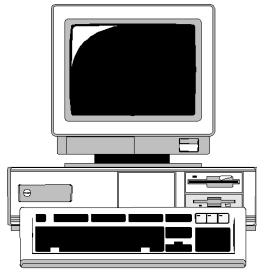


.
.
.
.
.
thesiger.cs.ucla.edu	131.178.192.144
.
.
.
.
.

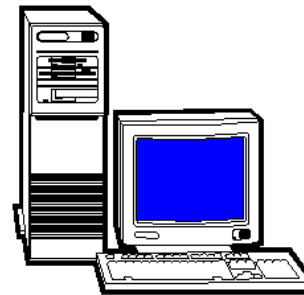
That wasn't very nice of him!

DNS Database Corruption

lookup thesiger.cs.ucla.edu



answer 97.22.101.53



...	...
...	...
...	...
...	...
...	...
thesiger.cs.ucla.edu	97.22.101.53
...	...
...	...
...	...
...	...
...	...

The DNSSEC Solution

- Sign the translations
- Who does the signing?
 - The server doing the response?
 - Or the server that “owns” the namespace in question?
- DNSSEC uses the latter solution

Implications of the DNSSEC Solution

- DNS databases must store signatures of resource records
- There must be a way of checking the signatures
- The protocol must allow signatures to be returned

Checking the Signature

- Basically, use certificates to validate public keys for namespaces
- Who signs the certificates?
 - The entity controlling the higher level namespace
- This implies a hierarchical solution

The DNSSEC Signing Hierarchy

- In principle, ICANN signs for itself and for top level domains (TLDs)
 - Like .com, .edu, country codes, etc.
- Each TLD signs for domains under it
- Those domains sign for domains below them
- And so on down

An Example

- Who signs the translation for thesiger.cs.ucla.edu to 131.179.192.144?
- The UCLA CS DNS server
- How does someone know that's the right server to sign?
- Because the UCLA server says so
 - Securely, with signatures
- The edu server verifies the UCLA server's signature
- Ultimately, hierarchical signatures leading up to ICANN's attestation of who controls the edu namespace
- Where do you keep that information?
 - In DNS databases

Using DNSSEC

- To be really secure, you must check signatures yourself
- Next best is to have a really trusted authority check the signatures
 - And to have secure, authenticated communications between trusted authority and you

A Major Issue

- When you look up something like `cs.ucla.edu`, you get back a signed record
- What if you look up a name that doesn't exist?
- How can you get a signed record for every possible non-existent name?

The DNSSEC Solution

- Names are alphabetically orderable
- Between any two names that exist, there are a bunch of names that don't
- Sign the whole range of non-existent names
- If someone looks one up, give them the range signature

For Example,

•
•
•



lasr.cs.ucla.edu	131.179.192.136	
pelican.cs.ucla.edu	131.179.128.17	
pelican.cs.ucla.edu	131.179.128.16	
pelican.cs.ucla.edu	131.179.128.17	
toucan.cs.ucla.edu	131.179.128.16	

•
•
•

You get
authoritative
information that
the name isn't
assigned

Foils spoofing
attacks

> host last.cs.ucla.edu

Status of DNSSEC

- Working implementations available
- In use in some places
- Heavily promoted
 - First by DARPA
 - Now by DHS
- Beginning to get out there

Status of DNSSEC Deployment

- ICANN has signed the root
 - Over 1100 TLDs have signed
 - Including .com, .gov, .edu, .org, .net
 - Not everyone below has signed, though
- Many “islands” of DNSSEC signatures
 - Signing for themselves and those below them
 - In most cases, just for themselves
- Utility depends on end machines checking signatures

Using DNSSEC

- Actually installing and using DNSSEC not quite as easy as it sounds
- Lots of complexities down in the weeds
- Particularly hard for domains with lots of churn in their namespace
 - Every new name requires big changes to what gets signed

Other DNS Security Solutions

- Encrypt communications with DNS servers
 - Prevents DNS cache poisoning
 - But assumes that DNS server already has right record
- Ask multiple servers
 - Majority rules or require consensus

Conclusion

- Correct Internet behavior depends on a few key technologies
 - Especially routing and DNS
- Initial (still popular) implementations of those technologies are not secure
- Work is ongoing on improving their security