Administration of Operating Systems

DNS & Bind

Chapter 24

December 01, 2011

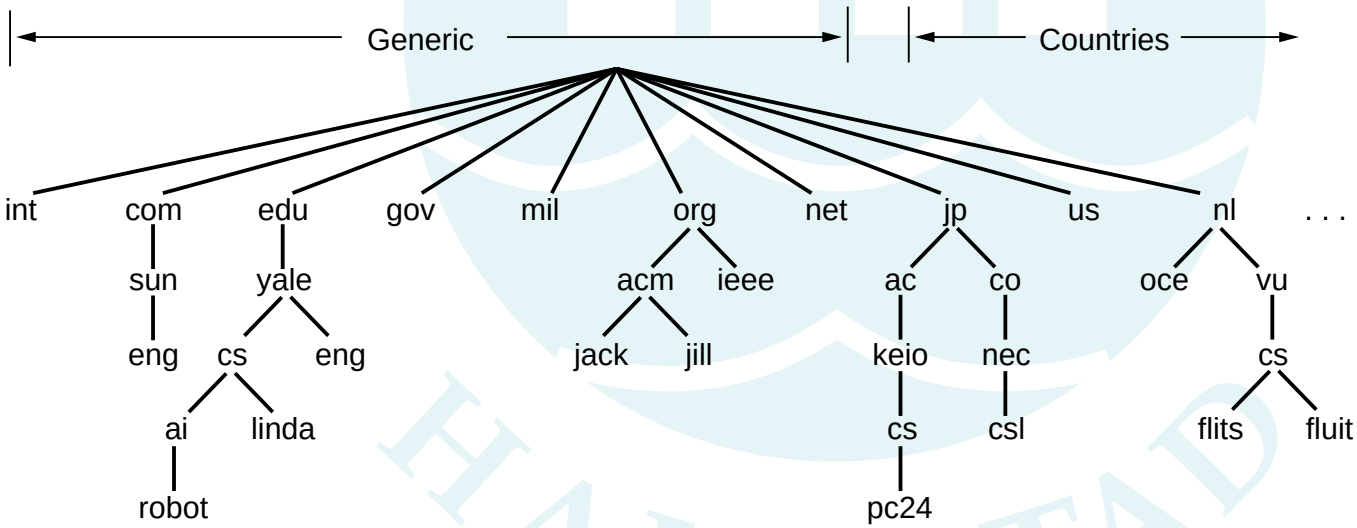
Intelligent Systems Lab

DNS

- Domain Name System
- RFC 1034 & 1035 and over 50 others
- Database for mapping human-readable domain names to corresponding IP addresses
  - hierarchical
  - distributed
- Uses TCP and/or UDP
  - application layer protocol
- Originally, in ARPANET, there was a file listing all names and their IP addresses
  - obviously, this does not scale too well

DNS Hierarchy

- Top-level domains
  - generic: com, edu, gov, int, mil, net, info, ...
  - countries: uk, se, pl, nu, no, ...
- Each domain is registered in superior domain
  - and manages its own sub-domains



DNS Resource Records

- Every domain has a number of different pieces of information associated with it
- Each *resource record* contains
  - domain name
  - class (“IN” for Internet)
  - type
  - value(s)
- *Start of Authority* type
  - time to live (in seconds)
  - serial number (for up-to-date checking)
  - authoritative nameserver
  - admin email address

FQDN

- Structure of DNS hierarchy tree is similar to Linux filesystem hierarchy tree
  - single root
  - absolute and relative paths

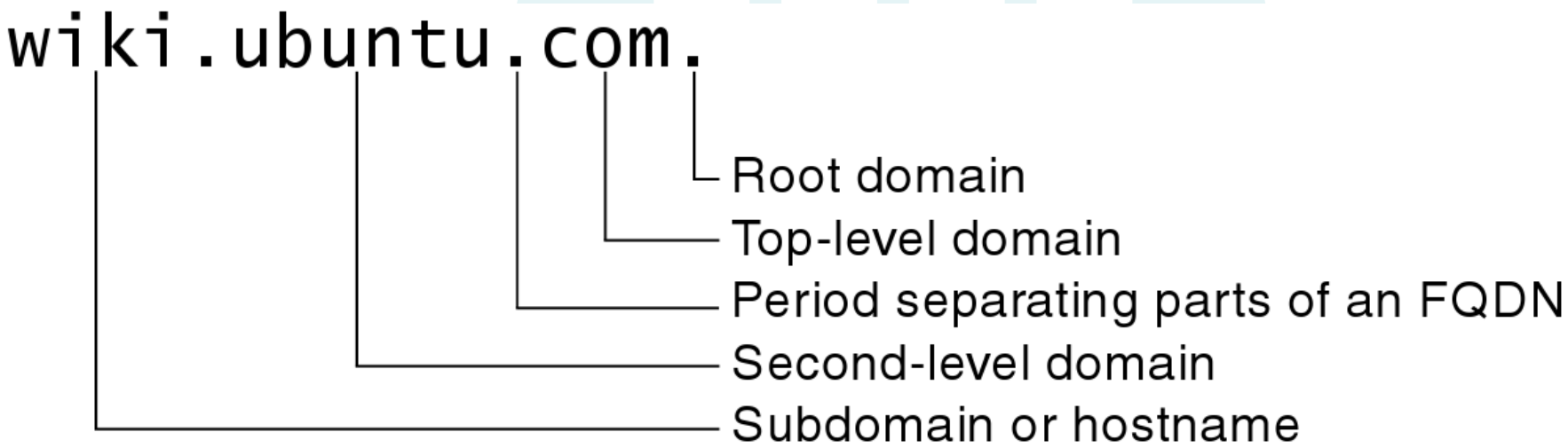


Figure 24-2 A fully qualified domain name (FQDN)

Record Types

- A — IPv4 Address
- AAAA type — IPv6 Address
- CNAME type — Canonical Name
  - hostname aliasing
- MX — Mail Exchange
- NS — Nameserver
- PTR — Pointer
  - reverse DNS queries
- TXT — Text
  - arbitrary information
- CERT, DNSKEY, IPSECKEY, TKEY, TSIG, ...
  - security-related records

DNS Resource Records

- From a client perspective, the most important type is the one containing the IP address
  - there can be several of those
  - DNS can be configured to cycle through them
  - DNS can provide both IPv4 and IPv6 address

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

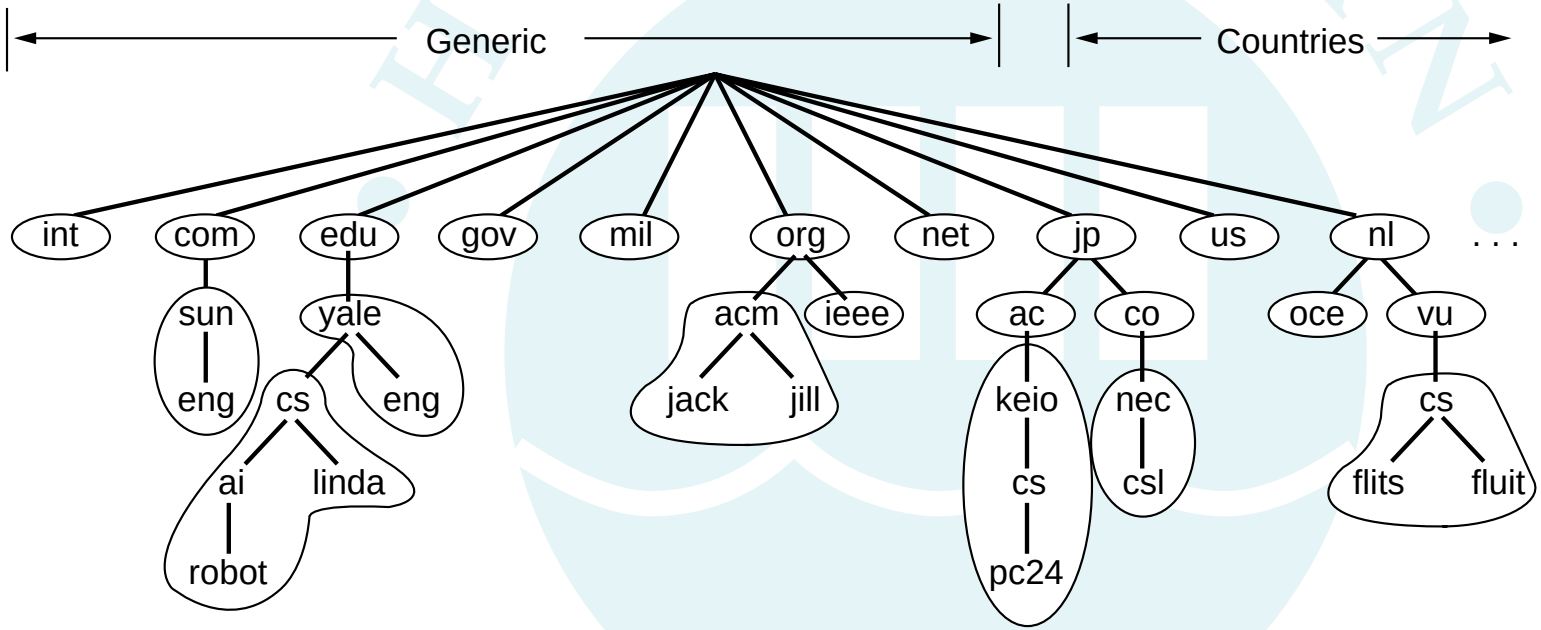
DNS Records Example

Intelligent Systems Lab

```
; Authoritative data for cs.vu.nl
cs.vu.nl.        86400  IN  SOA     star boss (952771,7200,7200,2419200,86400)
cs.vu.nl.        86400  IN  TXT     "Divisie Wiskunde en Informatica."
cs.vu.nl.        86400  IN  TXT     "Vrije Universiteit Amsterdam."
cs.vu.nl.        86400  IN  MX      1 zephyr.cs.vu.nl.
cs.vu.nl.        86400  IN  MX      2 top.cs.vu.nl.

flits.cs.vu.nl.  86400  IN  HINFO   Sun Unix
flits.cs.vu.nl.  86400  IN  A       130.37.16.112
flits.cs.vu.nl.  86400  IN  A       192.31.231.165
flits.cs.vu.nl.  86400  IN  MX      1 flits.cs.vu.nl.
flits.cs.vu.nl.  86400  IN  MX      2 zephyr.cs.vu.nl.
flits.cs.vu.nl.  86400  IN  MX      3 top.cs.vu.nl.
www.cs.vu.nl.    86400  IN  CNAME   star.cs.vu.nl
ftp.cs.vu.nl.    86400  IN  CNAME   zephyr.cs.vu.nl

rowboat                 IN  A       130.37.56.201
                        IN  MX      1 rowboat
                        IN  MX      2 zephyr
                        IN  HINFO   Sun Unix

little-sister           IN  A       130.37.62.23
                        IN  HINFO   Mac MacOS

laserjet                IN  A       192.31.231.216
                        IN  HINFO   "HP Laserjet IIISi" Proprietary
```

Domain Name Servers

- DNS space is divided into *zones*
  - each administered by a single DNS server



DNS Queries

- When DNS receives a query about its own domain, it answers with *authoritative record*
- Otherwise, requested data may be in cache
  - *non-authoritative response*
  - on all levels of the hierarchy
- Recursive query
  - server contacts the name who may know
  - usually starting with server for requested TLD
  - and propagate the request down the hierarchy
- Iterative query
  - reply with the address of next server to ask
  - less work for server, more work for client

Recursive Query

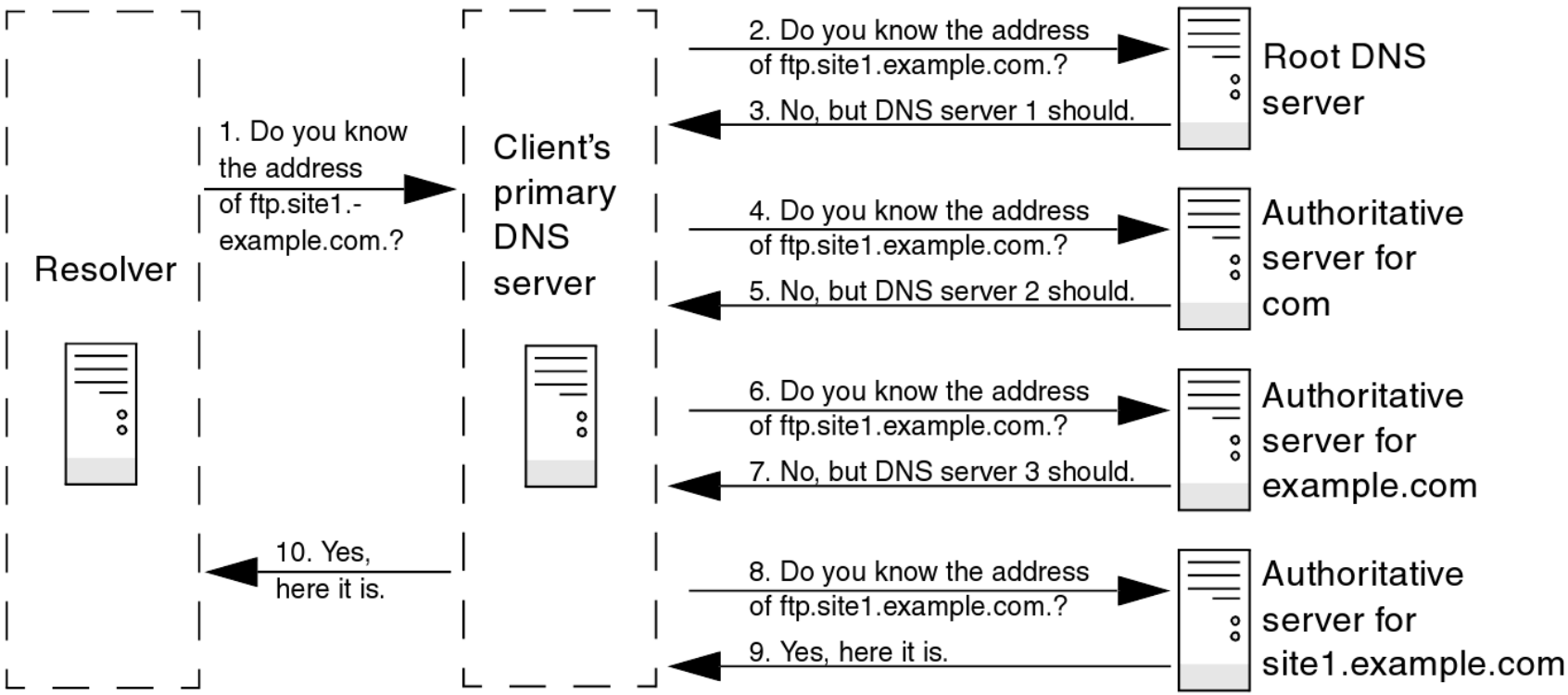


Figure 24-4 A recursive query that starts several iterative queries to find the answer

Intelligent Systems Lab

Glue Records

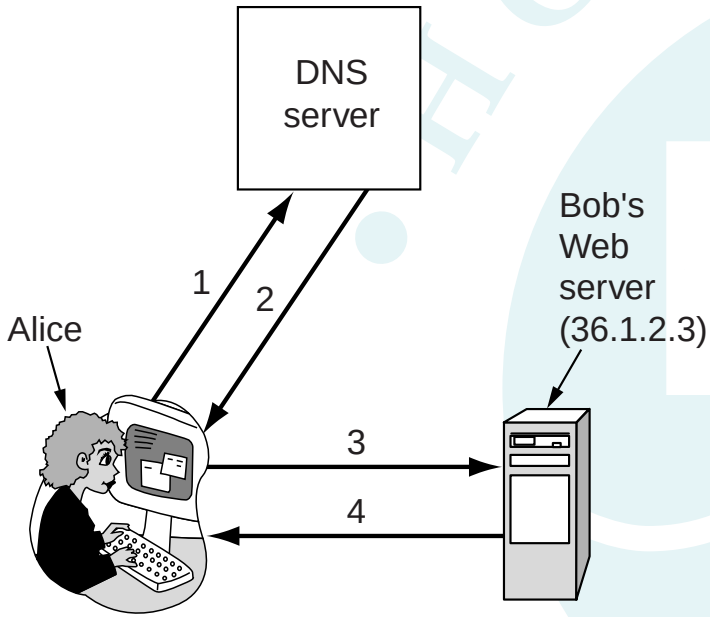
- Authoritative nameservers are usually located within the domains they serve
  - this could lead to circular dependencies
- Nameservers are identified by names
  - to continue resolving the query
  - client needs to contact one
  - it needs to find IP address
  - using DNS query
- Nameserver providing NS record includes “glue”
- IP address(es) for authoritative nameserver(s) mentioned in this NS record
  - “additional” section of the DNS response

Secure DNS

- Every DNS zone has a public/private key pair
  - all information coming from a DNS server must be signed with an appropriate private key
- DNSsec protocol includes mechanisms for
  - proving origin of the data
  - distribution of public keys
  - guaranteeing atomicity of transactions
  - authenticating the requests
- No confidentiality service
  - information is public anyway
- Security-aware servers must be able to interoperate with security-unaware ones

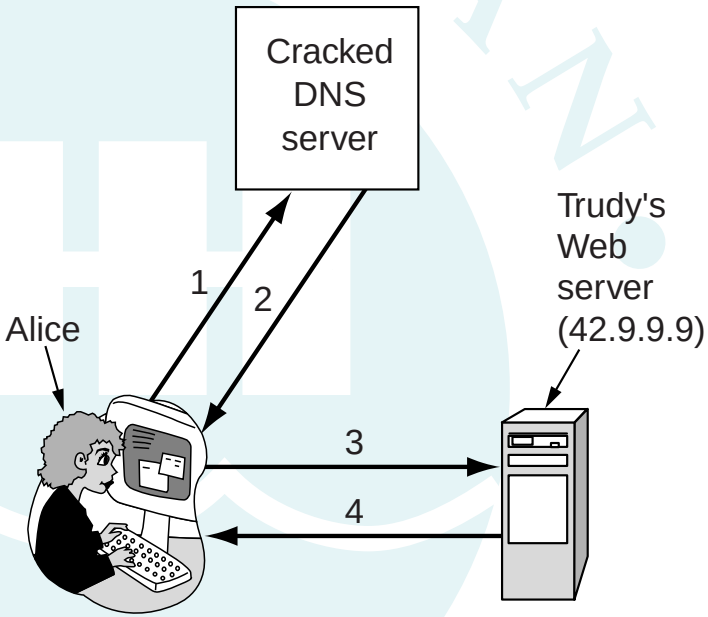
DNS Spoofing

- Trudy wants to convince Alice that Bob's IP address is 42.9.9.9 and not 36.1.2.3



1. Give me Bob's IP address
2. 36.1.2.3 (Bob's IP address)
3. GET index.html
4. Bob's home page

(a)



1. Give me Bob's IP address
2. 42.9.9.9 (Trudy's IP address)
3. GET index.html
4. Trudy's fake of Bob's home page

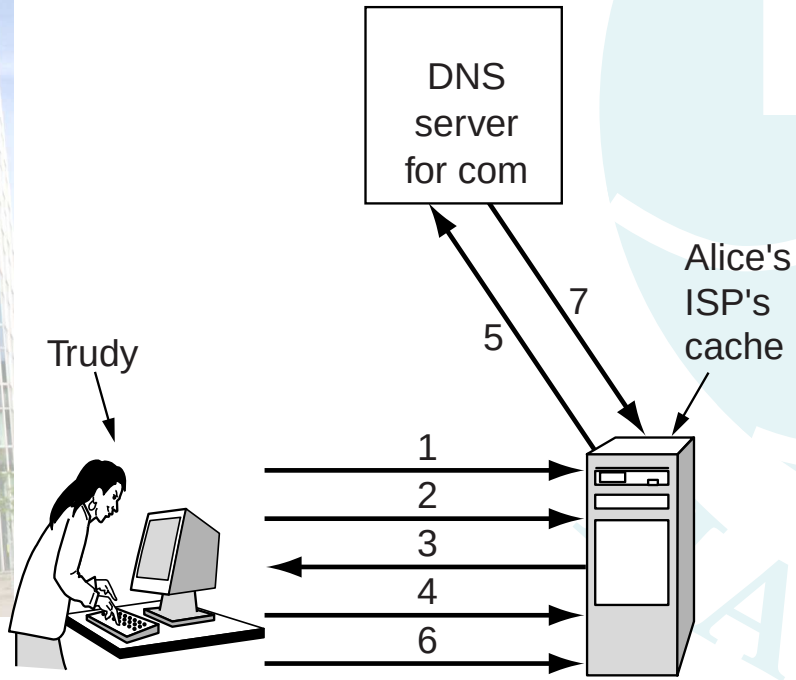
(b)

DNS Spoofing

- Main problem is that DNS spoofing is really easy
  - DNS uses UDP, so server has no way of knowing who really supplied an answer
  - and source IP number in UDP can be fake
- A scenario can be something like this
  - Trudy sends a request for Bob's IP address to DNS server belonging to Alice
    - it forwards the query to the top-level server
  - then, Trudy sends a *false* reply to this query
    - forging an IP address so it seems to be coming from the top-level DNS server
  - Alice's server will install false IP address in its cache and provide it whenever asked about Bob

DNS Spoofing

- Actually, DNS requests carry *sequence numbers* which responses need to match
  - Trudy needs server's current sequence number
  - but getting that is not difficult, even if Trudy cannot eavesdrop on the traffic!



1. Look up foobar.trudy-the-intruder.com
   (to force it into the ISP's cache)
2. Look up www.trudy-the-intruder.com
   (to get the ISP's next sequence number)
3. Request for www.trudy-the-intruder.com
   (Carrying the ISP's next sequence number, n)
4. Quick like a bunny, look up bob.com
   (to force the ISP to query the com server in step 5)
5. Legitimate query for bob.com with seq = n+1
6. Trudy's forged answer: Bob is 42.9.9.9, seq = n+1
7. Real answer (rejected, too late)

Intelligent Systems Lab

DNSsec

- All DNS records are grouped into units called Resource Record Sets
  - records with the same name, class and type
  - for example, several A records for a host with multiple IP addresses
  - RRSet is signed as a whole
- Each RRSet is hashed and signed with zone's private key (using, for example, SHA-1 and RSA)
  - since each RRSet contains its own signature, it can be cached even at untrustworthy servers
  - all that is needed is for each client to be able to acquire zone's public key

Offline mode

- DNSsec is designed in such a way that it is possible to keep zone's private key off-line
  - the only time it is actually needed is when some information changes and new records need to be signed
  - those new records can be moved to a machine not connected to the network and signed there
  - electronic security problem can be reduced to a physical security problem
- Crucial, since leaking the key for top-level domain would break the whole system
- It also speeds up operation, as no cryptography needs to be done while answering queries

DNSsec Record Types

- DNSsec introduces a number of additional record types related to security
- `KEY`, which holds the public key for a given zone and an algorithm for its use
- in the standard, the preferred algorithm is MD5, followed by RSA
- `SIG`, which holds a signed hash of all records in this particular RRSet
- `NXT`, for saying that there is no such name
  - “domain does not exist” is not safe against replay attacks if it is to be done offline
  - controversial, as this record allows to easily enumerate all names in a given zone

NSEC3

- One of the bigger problems with DNSsec is how to sign NXDOMAIN responses
  - no such domain exists
- It's not possible to pre-sign all possibilities
- Introduced NSEC response
  - "no hosts between `a.com` and `b.com`"
- Makes zone enumeration very easy
  - bad thing
- Partially fixed with NSEC3
  - DNS server publishes a hash function
  - and in which there are no valid *hashes*
  - both of those things can be signed

DNSsec Operation

- Clients are assumed to come pre-configured with public keys of all top-level domains
  - thus, Alice can ask *com* DNS server for RRSet corresponding to domain *bob.com*
  - which contains Bob's DNS server IP address *and* his zone's public key
- Reply from top-level will be signed, so there is no way Trudy can forge it
- Now Alice can *securely* get information directly from Bob's DNS server
- As an anti-replay security, a reply can contain (encrypted) copy of the request
  - thus binding the response to the query

Bind9

- Most widely used DNS server
  - there are several lighter alternatives
  - bind9 is a *de facto* standard
- Has a history of security vulnerabilities
  - getting better recently
- Master, slave or forwarder
- Multi-file configuration
  - `named.conf`
- Zone files
  - directives
  - SOA record
  - other records

named.conf

- Options
  - allow-query
  - allow-recursion
  - forward & forwarders
  - notify
  - recursion
- Zone clauses
  - file
  - masters
  - allow update
  - type: forward, hint, master, slave

Views

- Bind9 offers functionality of *views*
- Some responses for clients from local network
  - different ones from the Internet

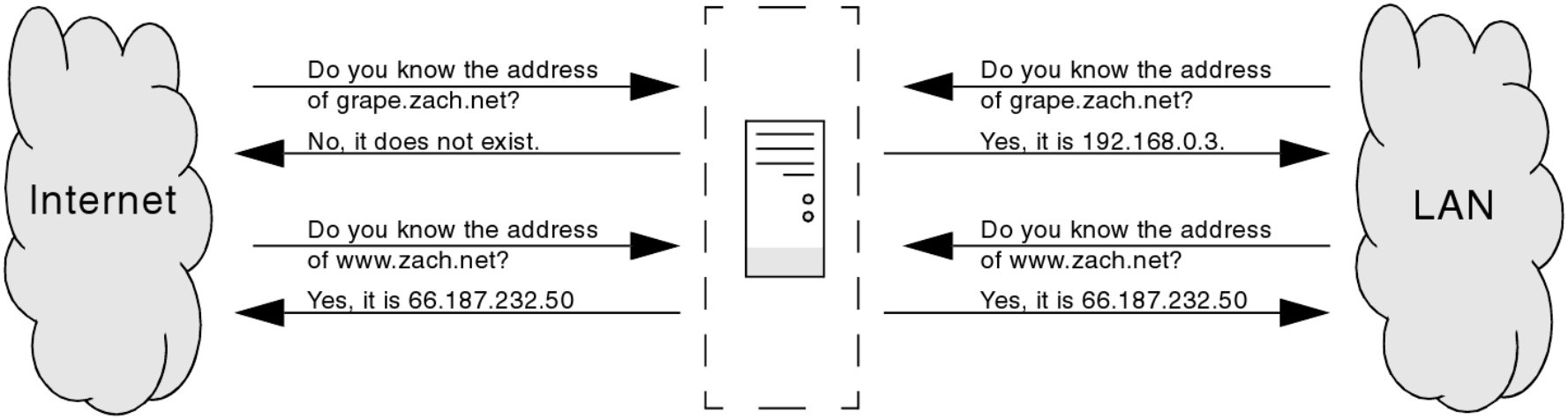


Figure 24-6 A split horizon DNS server

Questions?

Intelligent Systems Lab

https://192.168.5.5:10000/