DOMAIN NAME SYSTEM

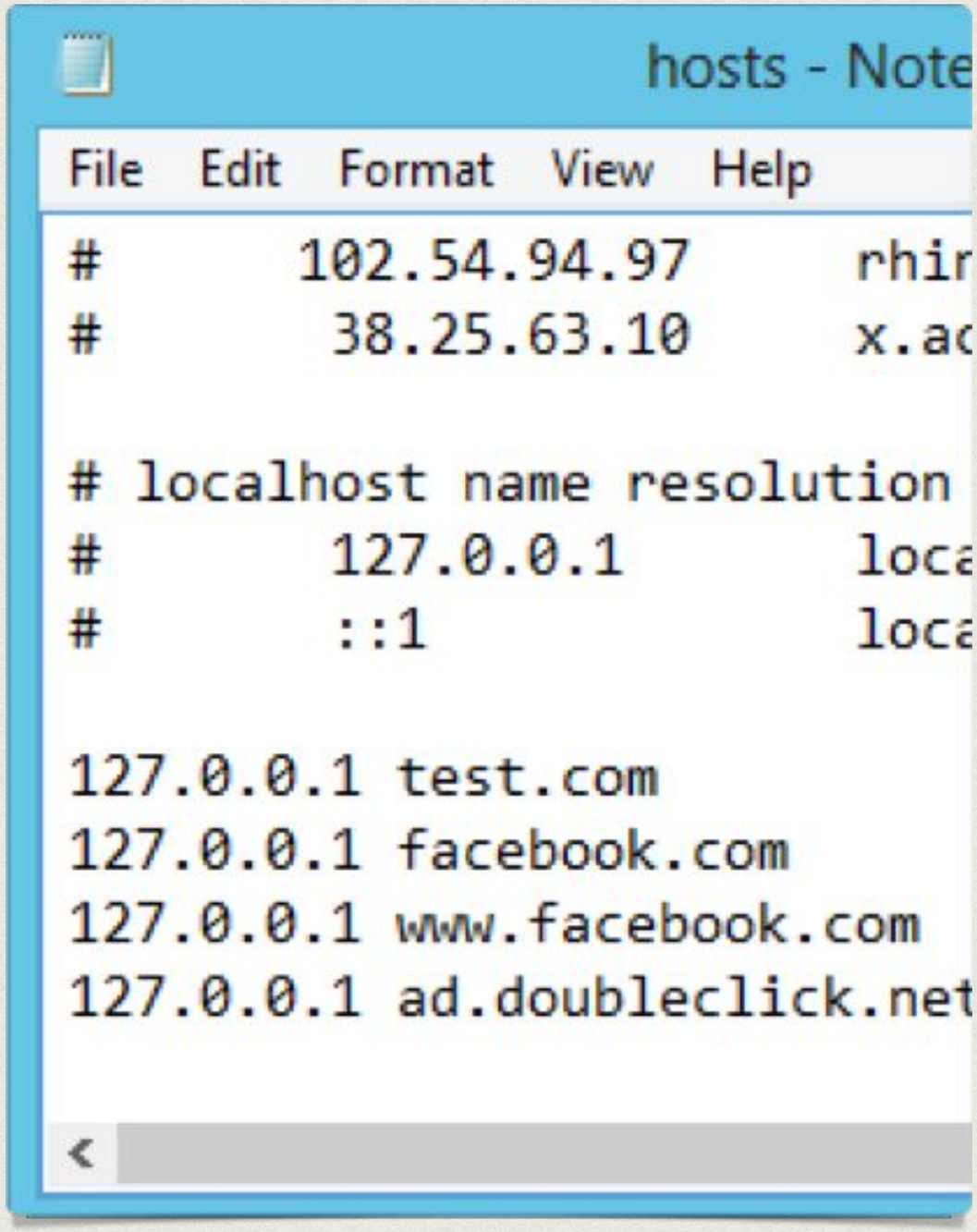
The Internet's Phone Book

NAME RESOLUTION

ONE IP TO MULTIPLE HOSTNAMES
(VIRTUAL HOSTING)

NO 1:1 MAPPING!

MULTIPLE IPS TO ONE HOSTNAME
(LOAD BALANCING)

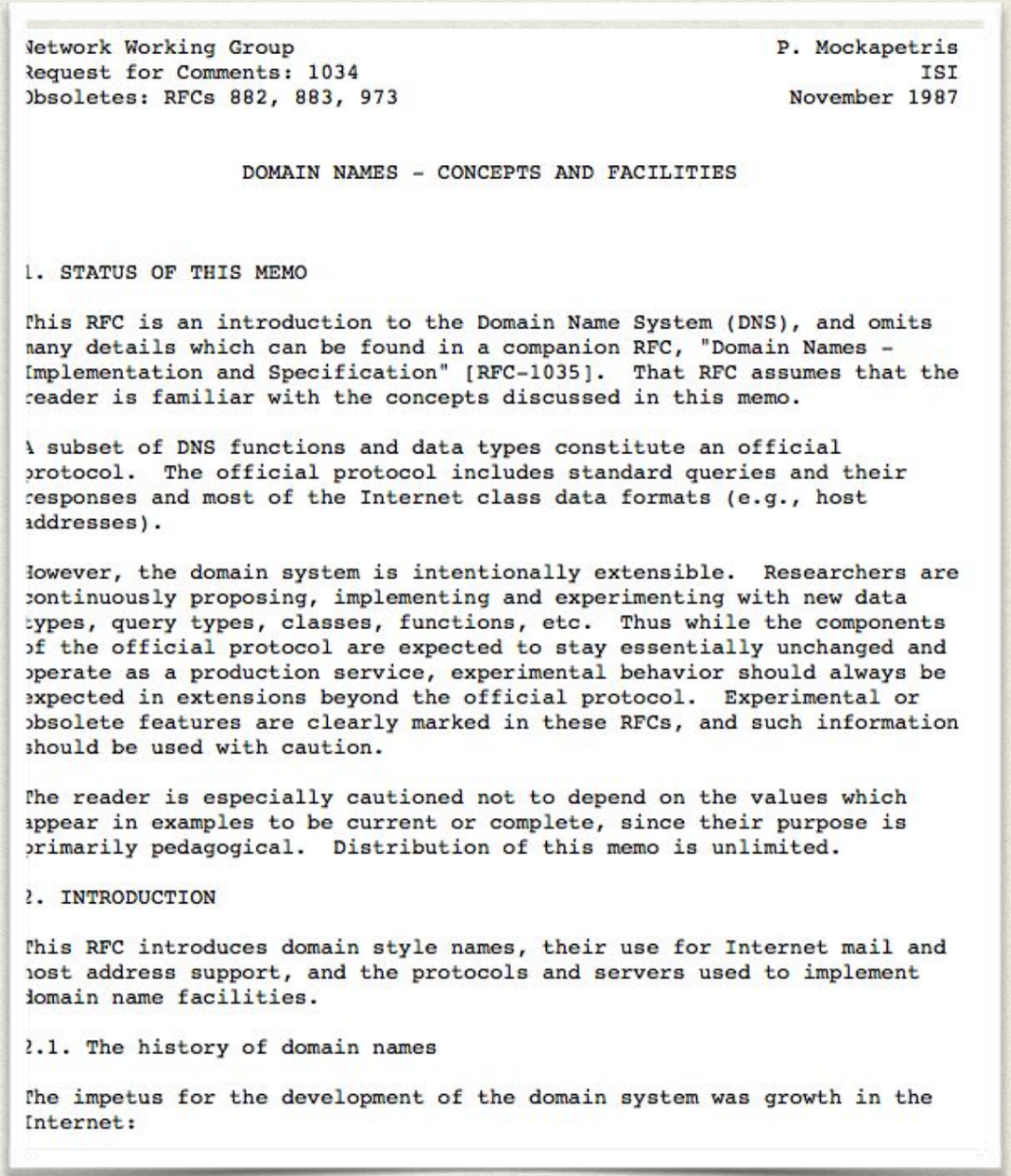


HOSTS FILE

Local Lookups

DNS

RFC1034 & RFC1035



Network Working Group — P. Mockapetris
Request for Comments: 1034 — ISI
Obsoletes: RFCs 882, 883, 973 — November 1987

DOMAIN NAMES - CONCEPTS AND FACILITIES

1. STATUS OF THIS MEMO

This RFC is an introduction to the Domain Name System (DNS), and omits many details which can be found in a companion RFC, "Domain Names - Implementation and Specification" [RFC-1035]. That RFC assumes that the reader is familiar with the concepts discussed in this memo.

A subset of DNS functions and data types constitute an official protocol. The official protocol includes standard queries and their responses and most of the Internet class data formats (e.g., host addresses).

However, the domain system is intentionally extensible. Researchers are continuously proposing, implementing and experimenting with new data types, query types, classes, functions, etc. Thus while the components of the official protocol are expected to stay essentially unchanged and operate as a production service, experimental behavior should always be expected in extensions beyond the official protocol. Experimental or obsolete features are clearly marked in these RFCs, and such information should be used with caution.

The reader is especially cautioned not to depend on the values which appear in examples to be current or complete, since their purpose is primarily pedagogical. Distribution of this memo is unlimited.

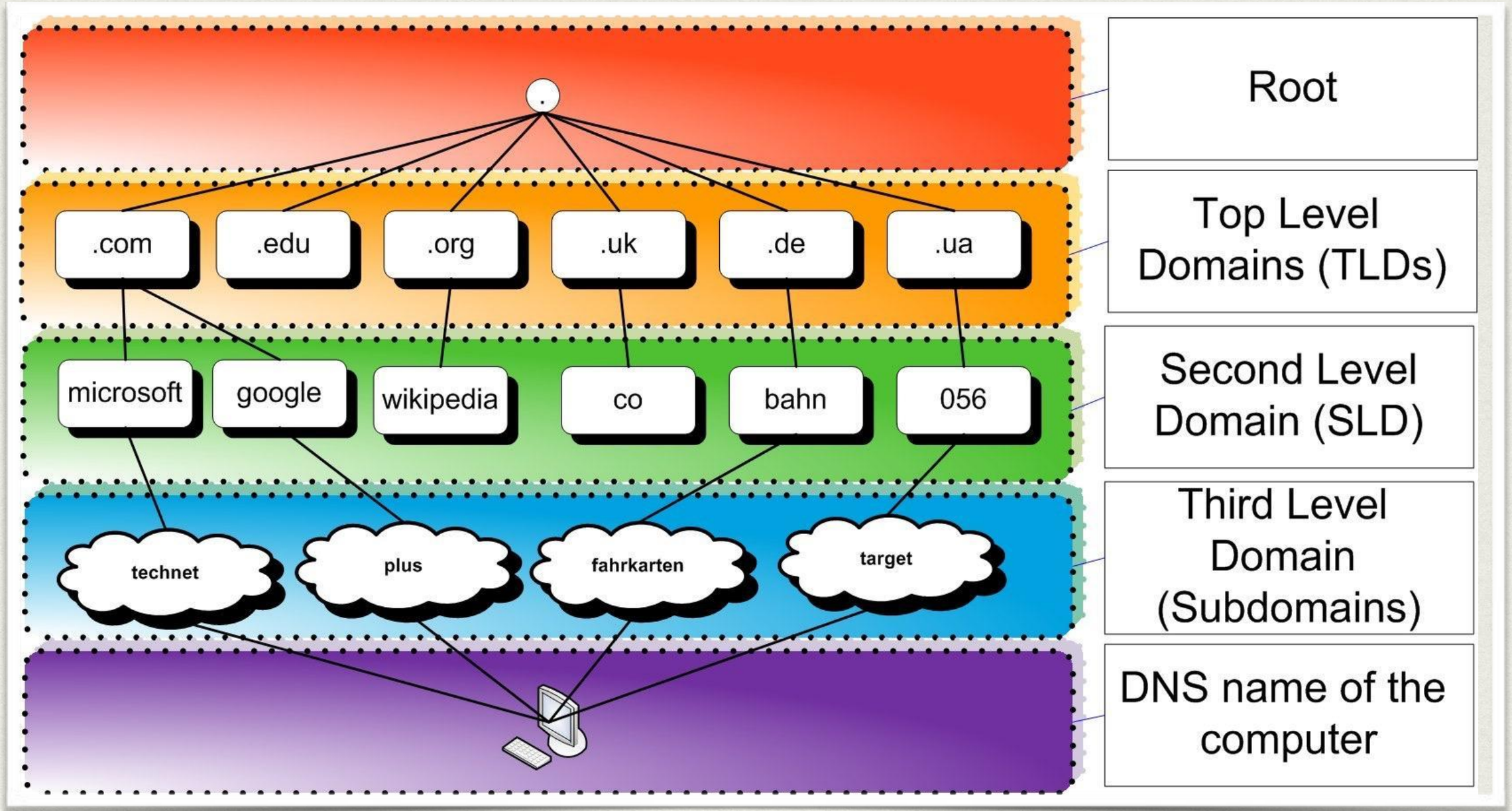
2. INTRODUCTION

This RFC introduces domain style names, their use for Internet mail and host address support, and the protocols and servers used to implement domain name facilities.

2.1. The history of domain names

The impetus for the development of the domain system was growth in the Internet:

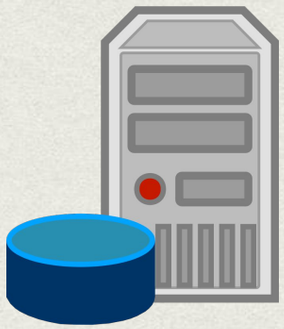
www.digipen.edu.



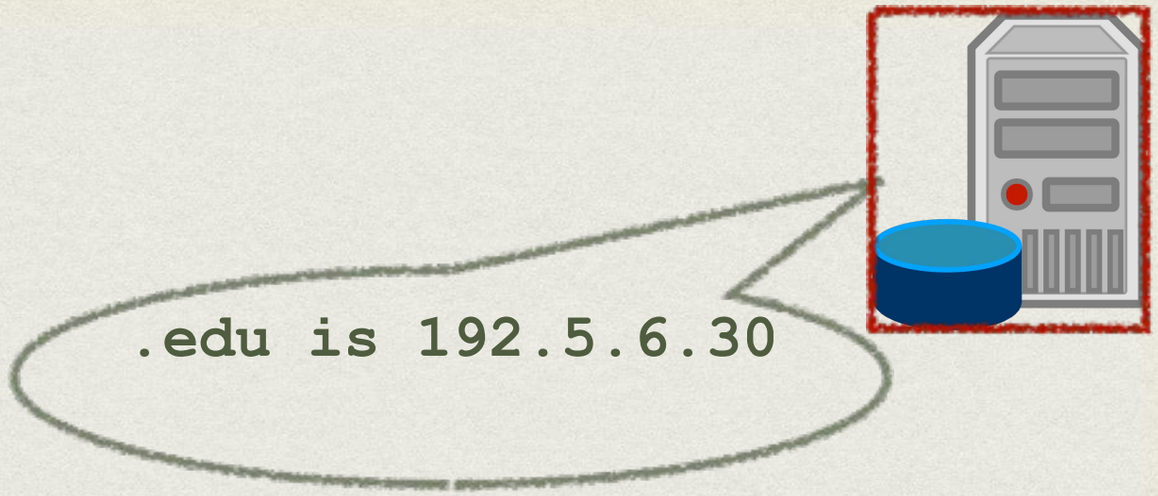
DNS RESOLUTION

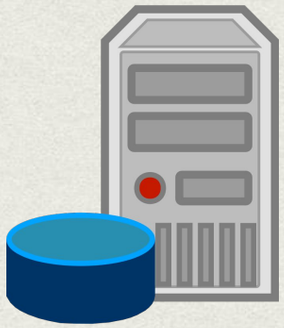
DNS RESOLUTION

Hey root: .edu?

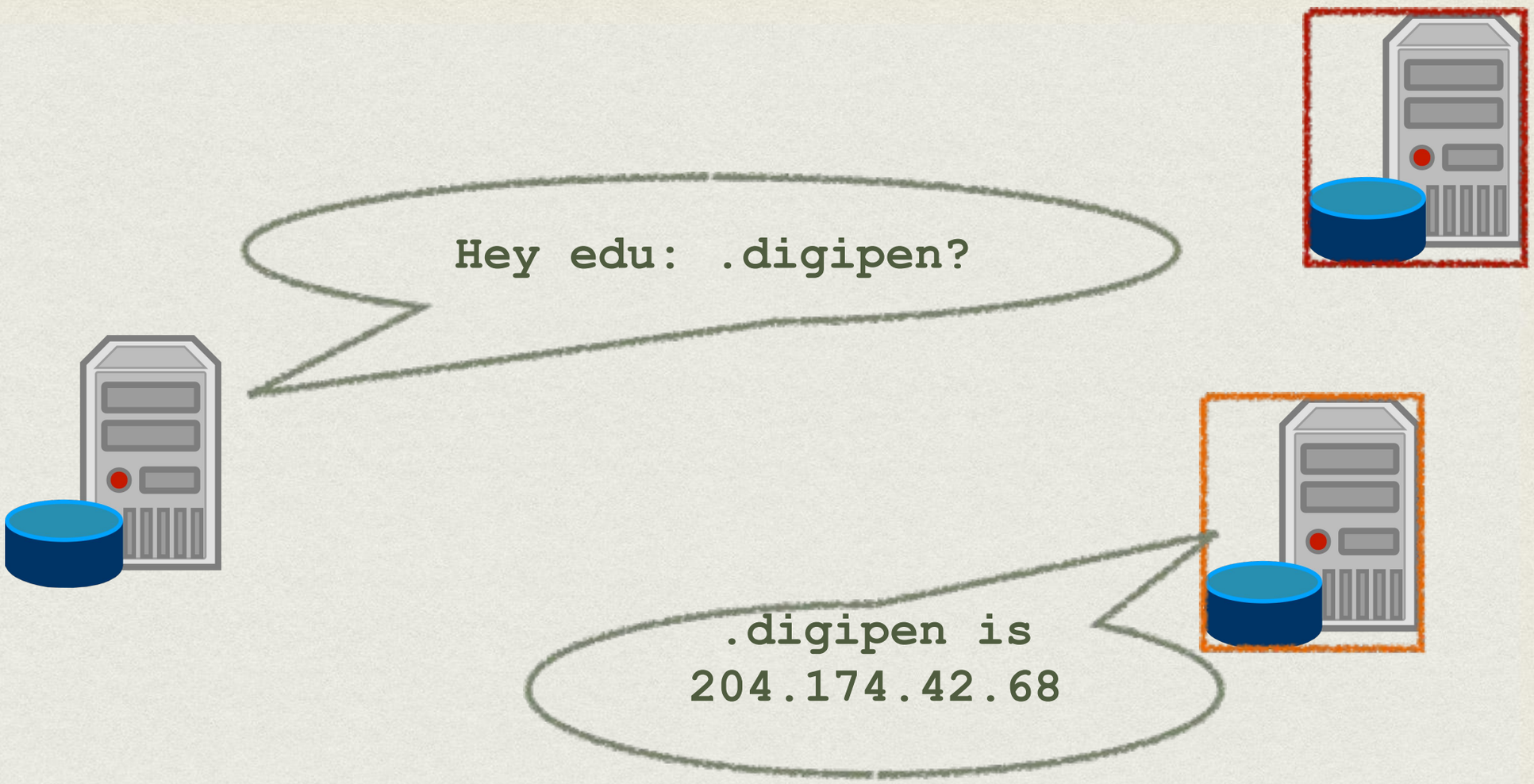


DNS RESOLUTION

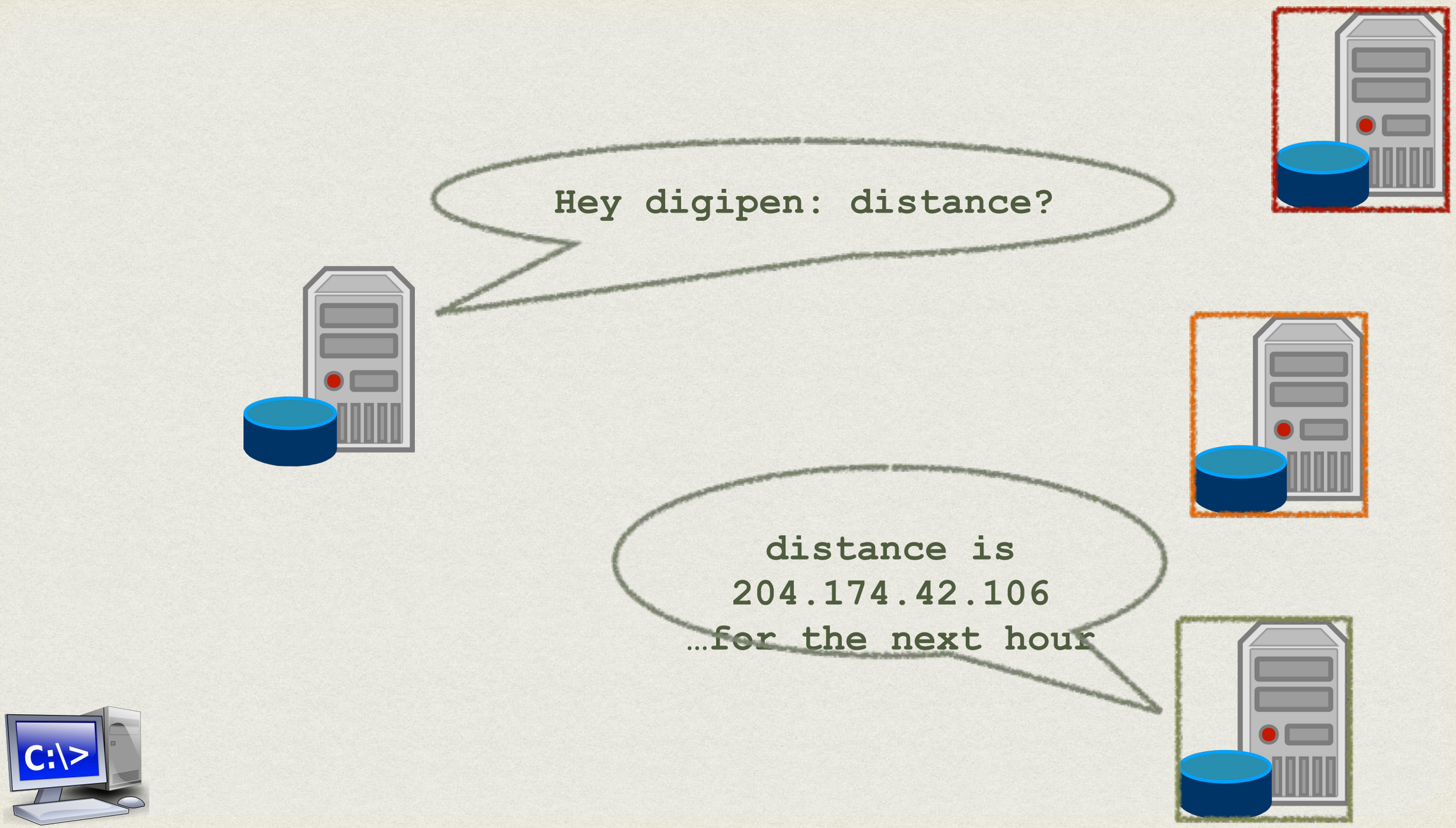




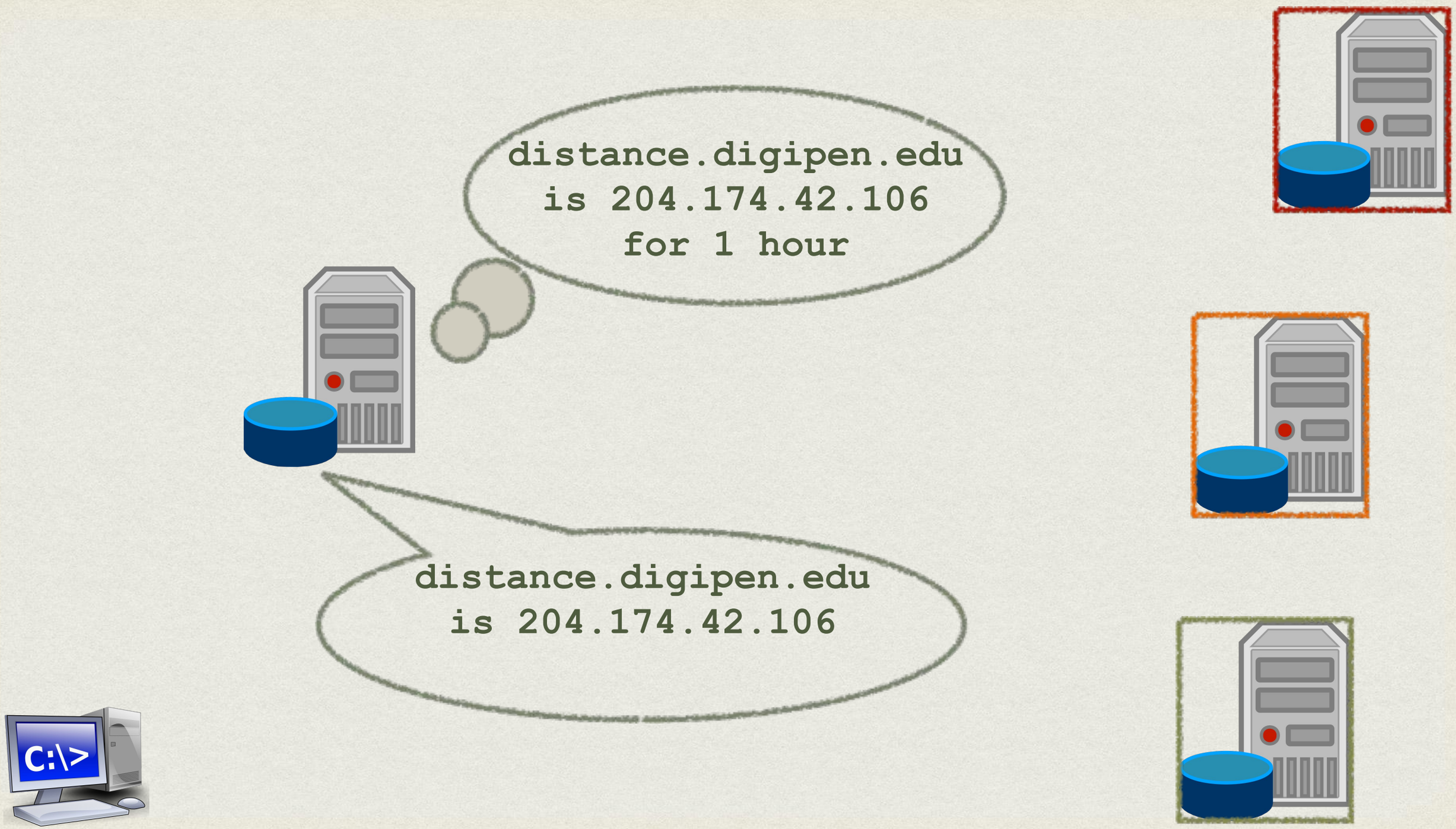
DNS RESOLUTION



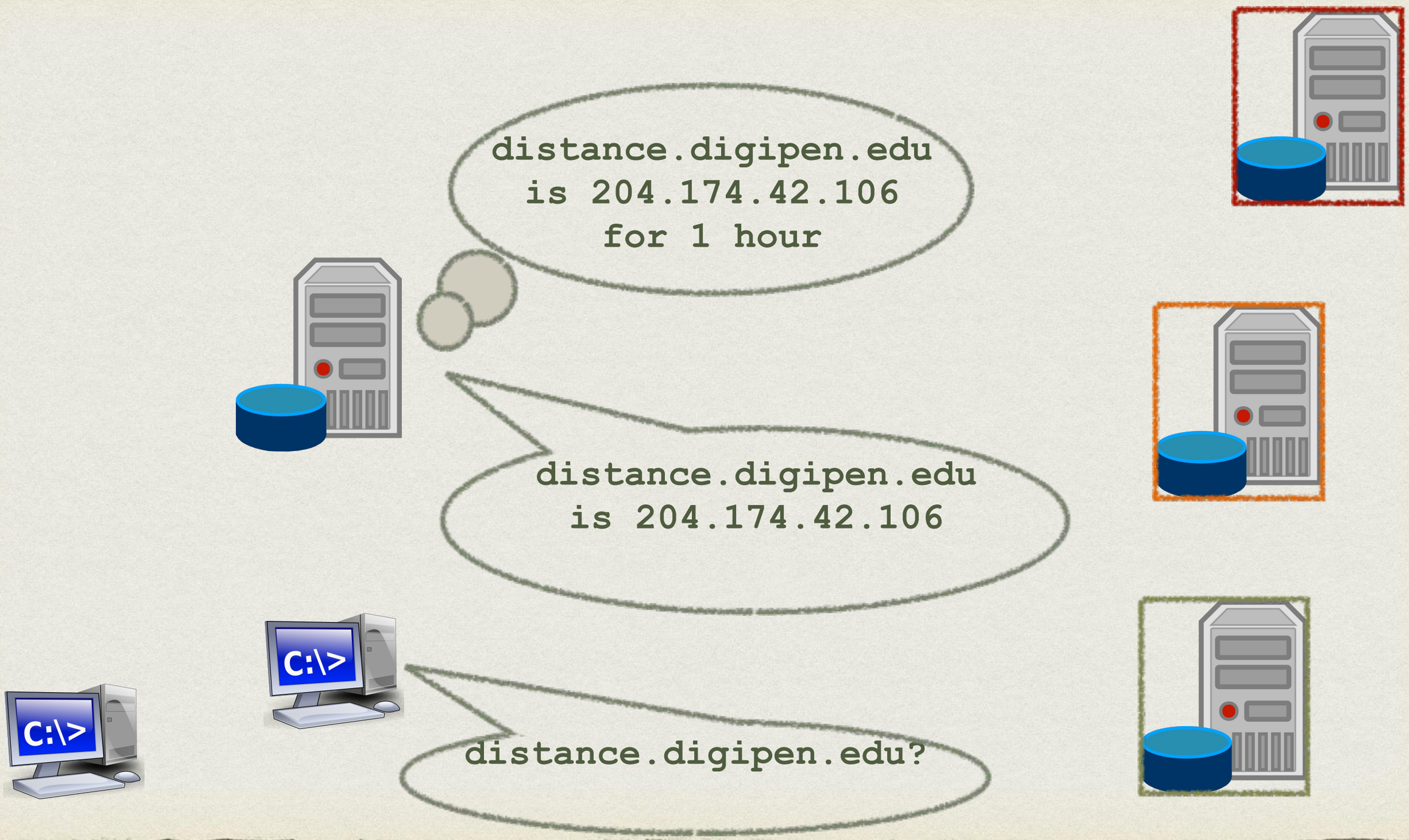
DNS RESOLUTION



DNS RESOLUTION



DNS RESOLUTION



NOT JUST NAMES

- A: Name to IP address
- CNAME: Name to another name
- MX: Mail servers for a domain
- SOA: “Start of Authority”
- TXT: “Here’s some random data”
- ...and many more at http://en.wikipedia.org/wiki/List_of_DNS_record_types

REVERSE LOOKUP

in-addr.ar

stebee — bash — 80×24

```
Jaegermac:~ stebee$ host google.com
google.com has address 173.194.33.1
google.com has address 173.194.33.8
google.com has address 173.194.33.4
google.com has address 173.194.33.2
google.com has address 173.194.33.6
google.com has address 173.194.33.0
google.com has address 173.194.33.5
google.com has address 173.194.33.9
google.com has address 173.194.33.7
google.com has address 173.194.33.14
google.com has address 173.194.33.3
google.com has IPv6 address 2607:f8b0:400a:800::1005
google.com mail is handled by 30 alt2.aspmx.l.google.com.
google.com mail is handled by 20 alt1.aspmx.l.google.com.
google.com mail is handled by 10 aspmx.l.google.com.
google.com mail is handled by 50 alt4.aspmx.l.google.com.
google.com mail is handled by 40 alt3.aspmx.l.google.com.
Jaegermac:~ stebee$ host 173.194.33.14
14.33.194.173.in-addr.arpa domain name pointer sea09s01-in-f14.1e100.net.
Jaegermac:~ stebee$
```

~~gethostbyname()~~

gethostbyaddr()

CREATING ADDRESSES WITH `getaddrinfo()`

```
sockaddr_in* CreateAddress(char* ip, int port)
{
  sockaddr_in* result =
(sockaddr_in*)calloc(sizeof(*result), 1);

  result->sin_family = AF_INET;
  result->sin_port = htons(port);

  if (ip == NULL)
    result->sin_addr.S_un.S_addr = INADDR_ANY;
  else
    result->sin_addr.S_un.S_addr = inet_addr(ip);

  return result;
  // Caller will be responsible for free()
}
```

CREATING ADDRESSES WITH `getaddrinfo()`

```
if (ip == NULL)
   result->sin_addr.S_un.S_addr = INADDR_ANY;
else
   result->sin_addr.S_un.S_addr = inet_addr(ip);

   return result;
   // Caller will be responsible for free()
}
```

CREATING ADDRESSES WITH `getaddrinfo()`

```
if (ip == NULL)
   result->sin_addr.S_un.S_addr = INADDR_ANY;
else
{
   result->sin_addr.S_un.S_addr = inet_addr(ip);

}

return result;
// Caller will be responsible for free()
}
```

CREATING ADDRESSES WITH `getaddrinfo()`

```
if (ip == NULL)
   result->sin_addr.S_un.S_addr = INADDR_ANY;
else
{
   result->sin_addr.S_un.S_addr = inet_addr(ip);
   if (result->sin_addr.S_un.S_addr == INADDR_NONE)
   {
      ADDRINFOA hint;
      PADDRINFOA* ppAddrs;
      memset(&hint, 0, sizeof(ADDRINFOA));
      hint.ai_family = AF_INET;
      int err = getaddrinfo(ip, NULL, &hint, ppAddrs);
      if (err != 0)
         return NULL;
      else
      {
```

CREATING ADDRESSES WITH `getaddrinfo()`

```
    PADDRINFOA* ppAddrs;
    memset(&hint, 0, sizeof(ADDRINFOA));
    hint.ai_family = AF_INET;
    int err = getaddrinfo(ip, NULL, &hint, ppAddrs);
    if (err != 0)
      return NULL;
    else
    {
      result->sin_addr =
        ((SOCKADDR_IN*)ppAddrs->ai_addr)->sin_addr;
      freeaddrinfo(ppAddrs);
    }
  }
}
return result;
// Caller will be responsible for free()
}
```

`getaddrinfo()` blocks!

NAME RESOLUTION

- Local override from HOSTS file
- Distributed hierarchical database
- Results are cached
- getaddrinfo() is ugly, blocking and unavoidable

DNS POISONING