Universisdad Industrial de Santander
Escuela de Sistemas

Redes de Computadoras
Capa de Red

Prof. Gilberto Díaz
gilberto.diaz@uis.edu.co

Capa de Transporte

- La capa de transporte no es una capa más, es el corazón de todos los protocolos de redes. Aquí se lleva a cabo el control de la transmisión y la gestión de errores

- **Objetivo:** El objetivo principal de esta capa es proporcionar servicios eficientes y confiables a sus usuarios (generalemente los procesos de la capa de aplicación)

Capa de Transporte

- La capa de transporte es generalmente implantada en el kernel de S.O, como un proceso o como una biblioteca de funciones.

- Al software que realiza las funciones de esta capa se le conoce como **Entidad de Transporte**

Capa de Transporte

- Esta capa hace más confiable la transmisión de información encargándose de la recuperación de eventos como paquetes perdidos o paquetes mal formados

- Gracias a la capa de transporte los programadores de aplicaciones pueden escribir código según un conjunto estándar de primitivas y hacer que esas aplicaciones funcionen en una amplia variedad de redes

Capa de Transporte

- De la misma forma como nosotros podemos encontrar dos tipos de servicios de red, también tenemos dos tipos de servicios de transporte:
  - Orientado a conexión
  - No orientado a conexión

Capa de Transporte

- Para proporcionar el servicio de transporte se debe contar con ese conjunto de primitivas las cuales conforman la interfaz de la capa. Esta interfaz soporta los dos tipos de servicios
  - Orientado a conexión: (*streams)* donde se provee un servicio confiable
  - No orientado a conexión: (*datagram*) aqui no hay gestión de errores

Capa de Transporte

Berkeley Sockets

Es la combianación de una dirección IP y un puerto. Fueron desarrollados en la Universidad de Berkeley en 1983 y son el mecanismo más ampliamente utilizado para implantar aplicaciones de red

socket	crea un nuevo punto de comunicación
bind:	conecta el socket a la dirección local
listen:	anuncia que acepta conexiones
accept:	bloquea el llamador hasta que un intento de conexión arriba
connect:	intenta activamente establecer una conexión
send:	envía datos a través de la conexión
receive:	recibe datos a través de la conexión
close:	cierra la conexión

Capa de Transporte

Elementos Involucrados

- Direccionamiento
- Establecimiento de la conexión
- Cerrando la conexión
- Control del Flujo y buffers
- Multiplexación
- Recuperación de fallas

Capa de Transporte

En Internet se utilizan dos protocolos principales en la capa de transporte

- User Datagram Protocol (UDP): Dedicado a servicios no orientados a conexión. Es básicamente el mismo protocolo IP sólo que en la cabecera tiene información adicional

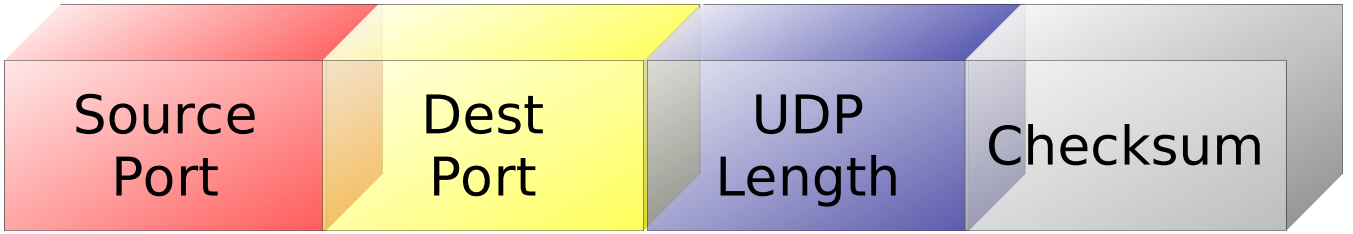
- Transmission Control Protocol (TCP): Dedicado a servicios orientados a conexión y proporciona mecanismos para establecer conexiones confiables

Capa de Transporte

User Datagram Protocol (UDP)

- UDP proporciona a las aplicaciones una forma de enviar datagramas IP encapsulados sin la necesidad de establecer primero una conexión.

- Los PDUs utilizados en UDP se denominan segmentos cuya cabecera es de 8 bytes. El *payload* es el contenido del paquete

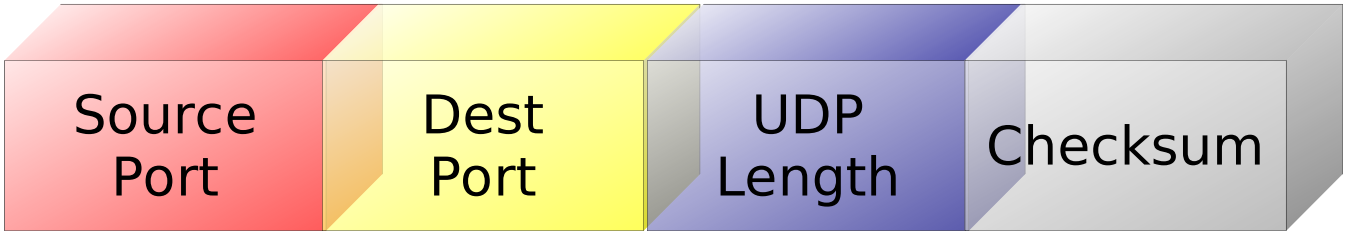


Capa de Transporte

User Datagram Protocol (UDP)

- El puerto de origen (Source port) es necesario cuando se necesita enviar una respuesta al nodo de origen

- El campo de longitud incluye los 8 bytes de la cabecera



Capa de Transporte

User Datagram Protocol (UDP)

- UDP no realiza tareas de:
  - Control de flujo
  - Control de errores
  - Retransmisión de paquetes mal formados o perdidos

Capa de Transporte

User Datagram Protocol (UDP)

- Sin embargo, UDP es útil en situaciones donde un cliente hace una solicitud corta y el servidor da una respuesta igualmente corta.

- Si el paquete se pierde simplemente se hace un reintento al transcurrir el *timeout*

- Esto es mucho más sencillo y no hay necesidad de mensajes adicionales para el establecimiento de la conexión.

Capa de Transporte

Remote Procedure Call (RPC)

- Este es un protocolo que utiliza UDP como base
- En cierto sentido enviar un mensaje a una máquina remota y recibir una respuesta es como hacer un llamado a un procedimiento
- En ambos casos se comienza con algunos parámetros y se finaliza con algunos resultados

Capa de Transporte

Remote Procedure Call (RPC)

- Esto hace que las aplicaciones de red sean mucho más sencillas y fáciles de entender.

- Esta técnica fue sugerida por Birell y Nelson en 1984 permitiendo a los programas hacer llamados a procedimientos localizados en máquinas remotas.

- El procedimiento que hace el llamado es denominado cliente y el procedimiento que es llamado es denominado servidor

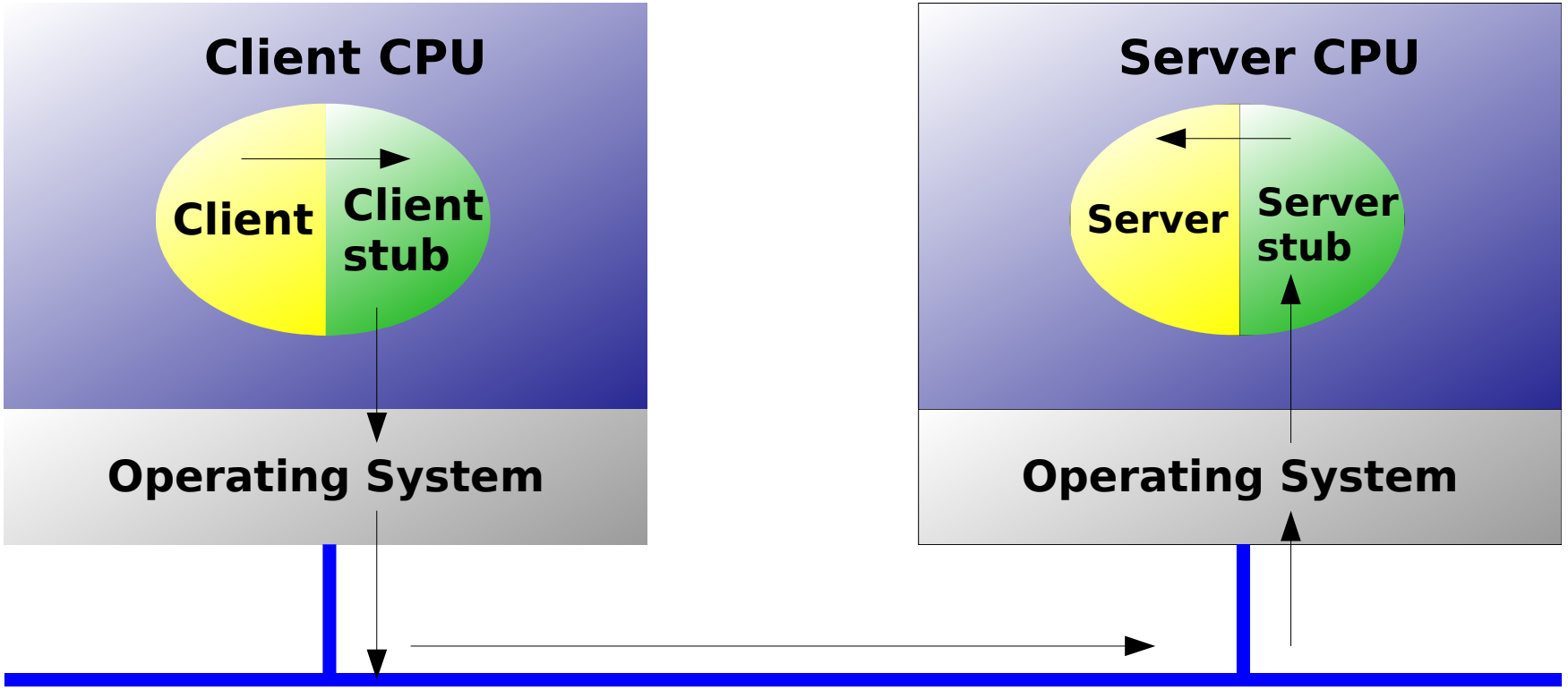
Capa de Transporte

Remote Procedure Call (RPC)

- Generalmente toda la complejidad se esconde utilizando bibliotecas con interfaces tanto para el servidor como para el cliente
  - *Client stub*
  - *Server stub*

Capa de Transporte

Remote Procedure Call (RPC)



Domain Name System (DNS)

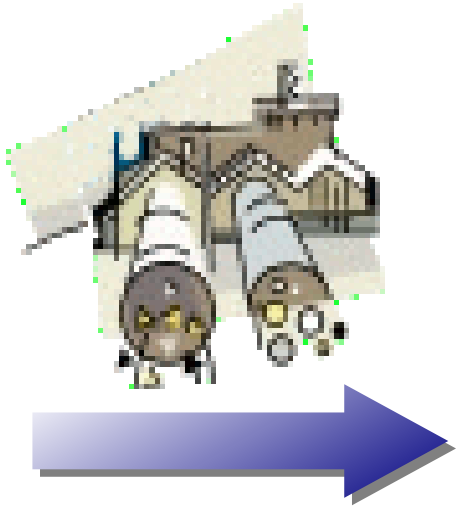
Es uno de los pilares fundamentales de Internet. Surgió como necesidad de extender el direccionamiento del correo electrónico al crecer ARPANET

Domain Name System (DNS)

Es responsable de traducir nombres de dominios a direcciones IP.

Nombre de dominio

www.uis.edu.co

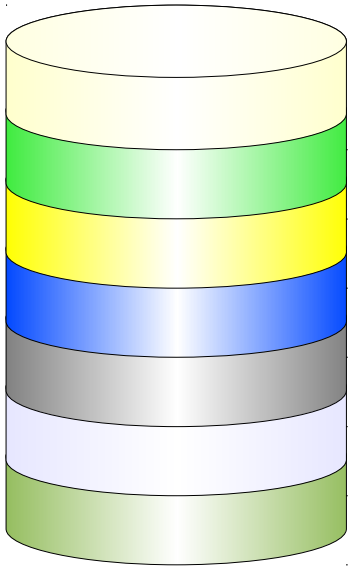


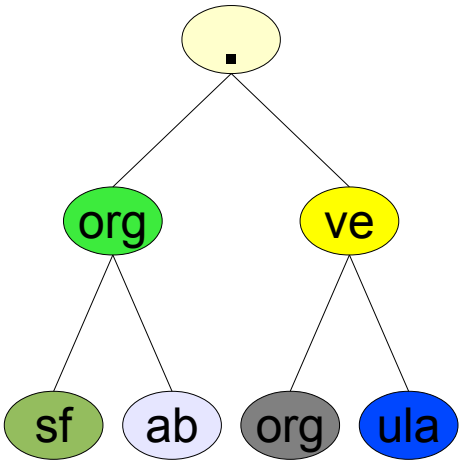
Dirección IP

192.168.3.1

Domain Name System (DNS)

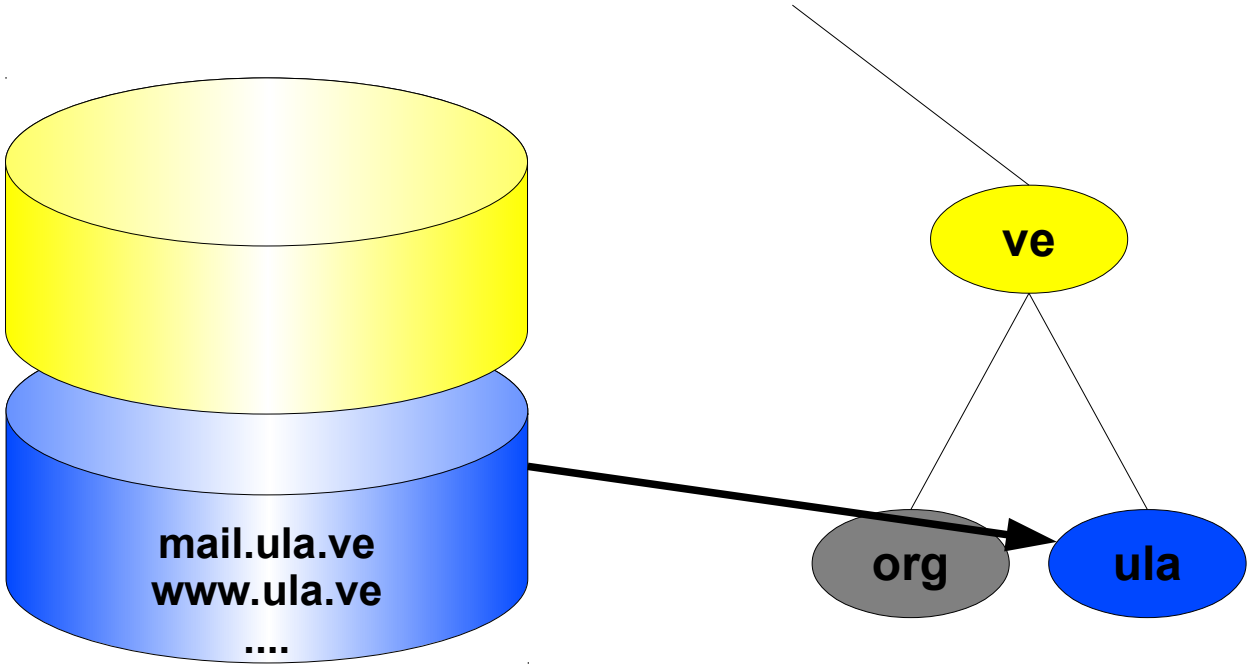
- Es una base de datos jerárquica distribuida a través de Internet.
- Los datos se almacenan en una estructura de árbol.
- Cada servidor de dominio almacena una porción de la base de datos.





Domain Name System (DNS)

Cada porción corresponde a la información de las máquinas pertenecientes a un dominio determinado.



Pero...

¿Qué es un dominio?

Como acepción inicial diremos que un dominio es una forma de ver al mundo como una jerarquía. La estructura más usual es la de árbol.

¿Qué es un dominio?

Los dominios han sido utilizados por años por sistemas como el servicio postal y el servicio telefónico.

Para comprender mejor este concepto haremos una reseña sobre estos dos sistemas.

El servicio postal ha dividido al mundo en:

- Paises
- Estados
- Ciudades, pueblos y caseríos
- Calles y avenidas
- Edificios, casas

Entonces nosotros tenemos direcciones de la siguiente forma:

Edif. General Masini, Calle 5 entre avs 18 y 19 Mérida edo Mérida - Venezuela

El servicio telefónico utiliza números pero los agrupa en códigos que representan a

- Paises
- Estados (operadoras)
- Identificación del individuo
- Extensión

De esta manera nosotros tenemos que para llamar a la Universidad de Los Andes

2401111
0274 2401111
58 274 2401111

- El servicio postal la parte más específica viene al principio (más a la izquierda)

- Mientras que en el servicio telefónico la parte más específica va al final (más a la derecha)

- El número telefónico a marcar depende de donde donde nos encontremos. Si estamos fuera del pais (119 274 2401111) o si es una llamada local (2401111). Esto es direcciones relativas.

- En el caso del servicio postal la dirección completa puede ser utilizada sin importar la ubicación del emisor (local o remota)

- ARPANET evolucionó utilizando un sistema de direcciones absolutas:
  - *usuario@máquina* funcionaba desde cualquier ubicación
- En el caso de UUCP se utilizaban direcciones relativas:
  - *máquina!usuario* funcionaba desde cualquier ubicación con un enlace directo a *máquina* y se tenía que enrutar el correo a través de la red hasta encontrar la máquina

- El formato *usuario@máquina* fue mucho más aceptado, sobre todo en aquellos sitios donde se corria algún tipo de software para correo.

- La estrategia utilizada era:
  - Buscar la *máquina* en una tabla y luego enviar el mensaje.

- El problema se hizo inmanejable cuando se tenian muchas máquinas, pues es muy complicado mantener una tabla con un gran número de registros.

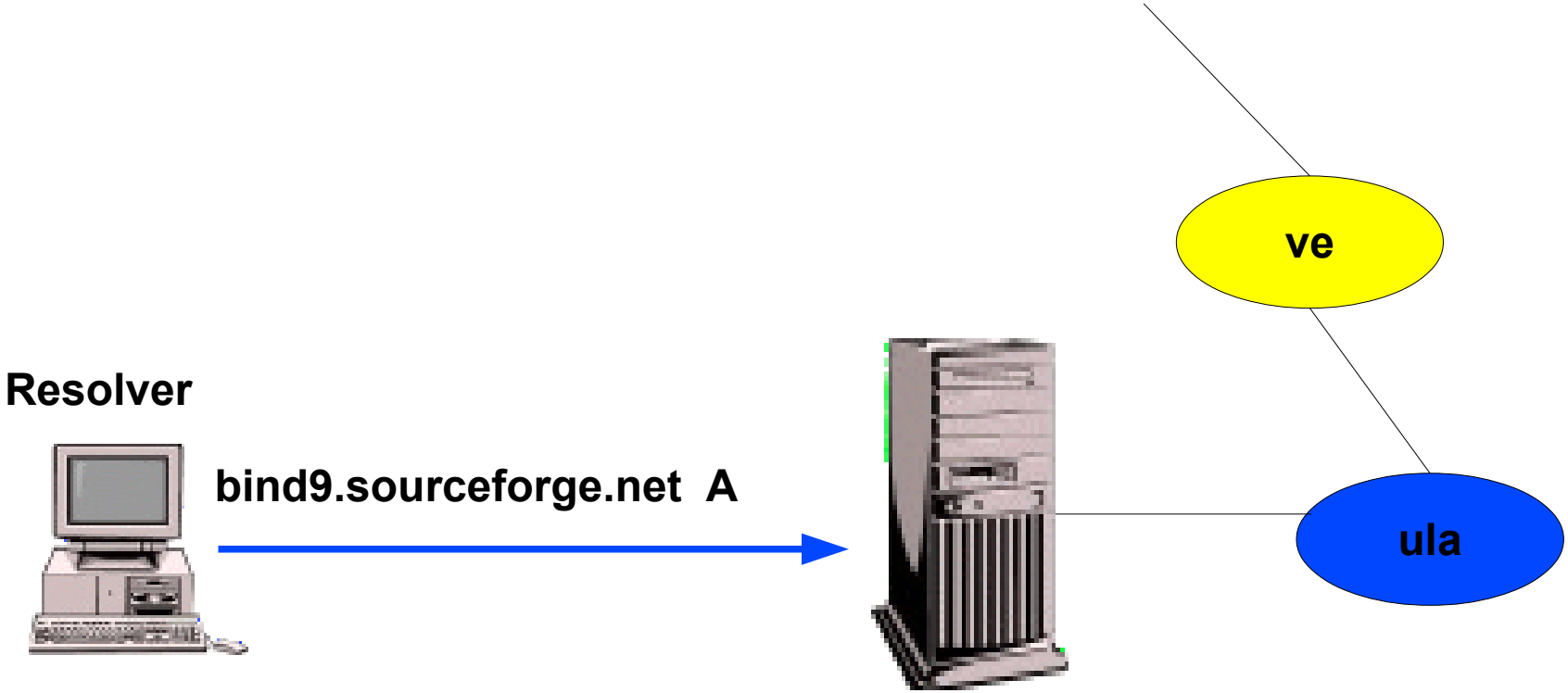
¿Qué es un dominio?

Luego de la revisión anterior podemos decir:

Un dominio es un sub árbol del árbol completo de DNS y representa todas las máquinas y nombres debajo de él.

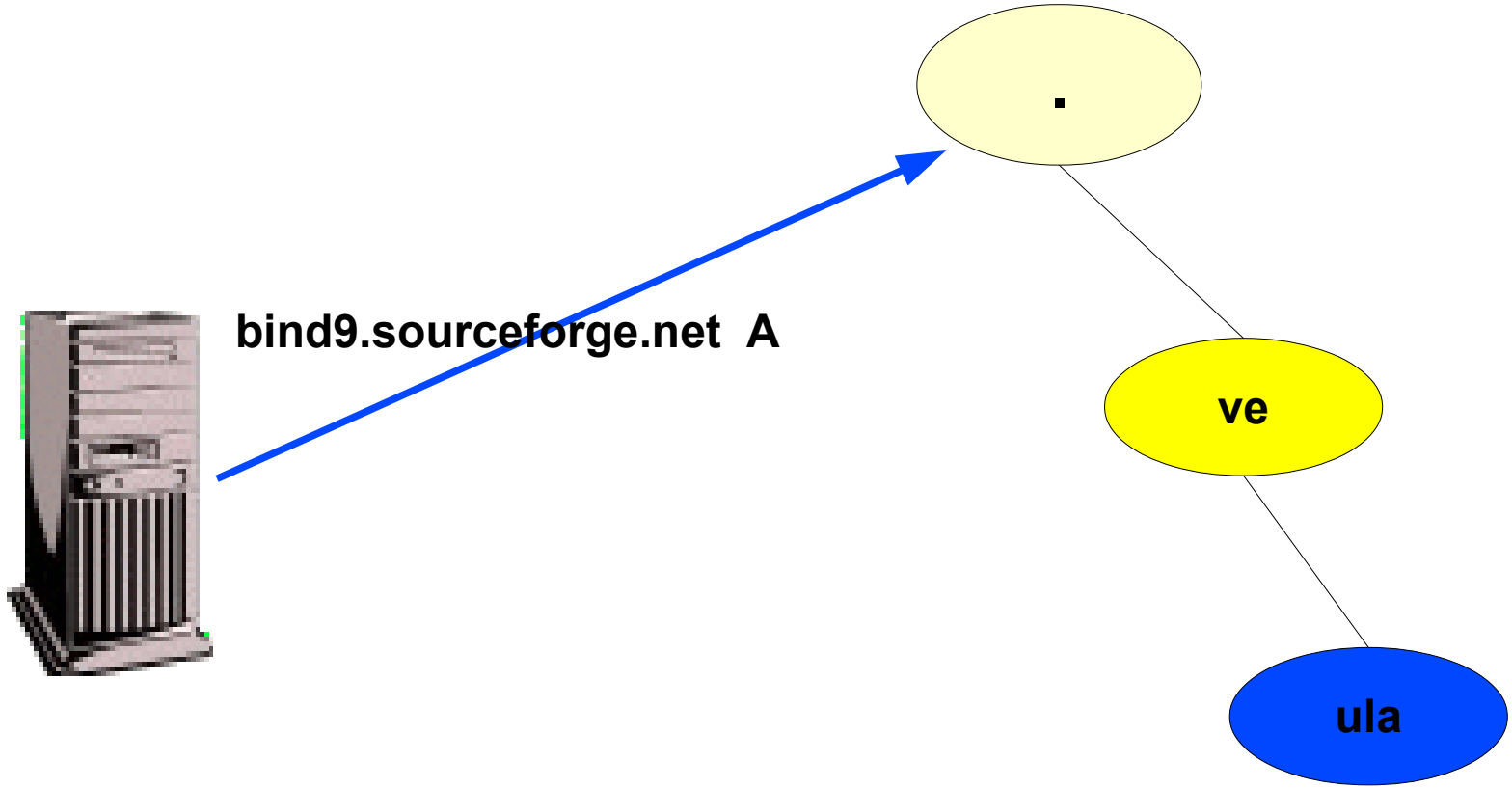
DNS trabaja bajo el principio de delegación

Debido a que los datos se encuentran distribuidos, cuando un servidor recibe una solicitud para resolver el nombre de un host que se encuentra fuera de su dominio, es probable que éste no tenga la información



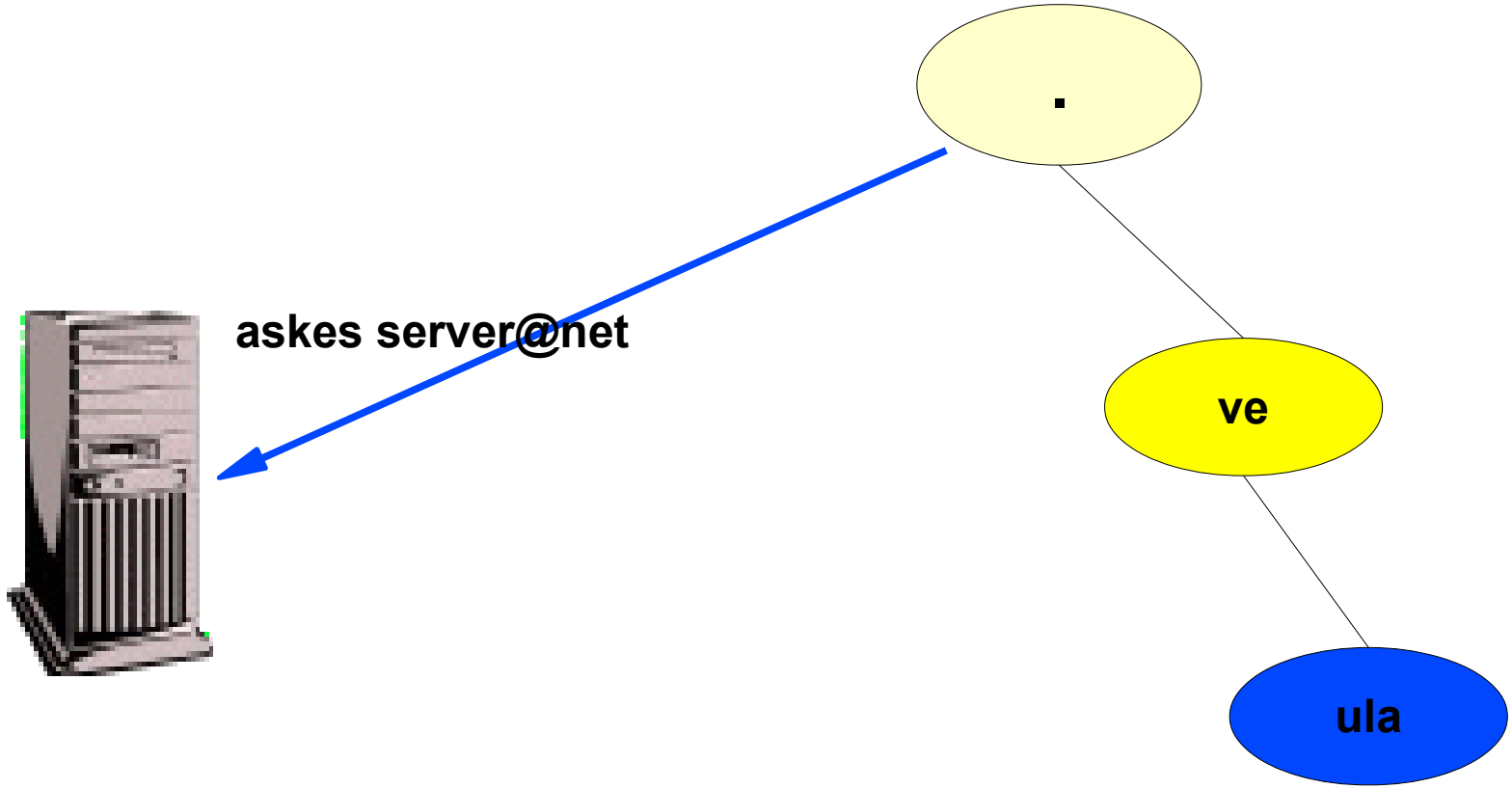
DNS trabaja bajo el principio de delegación

Para atender la solicitud lo único que necesita saber el servidor es como delegarle la solicitud a los servidores del dominio raiz (punto)



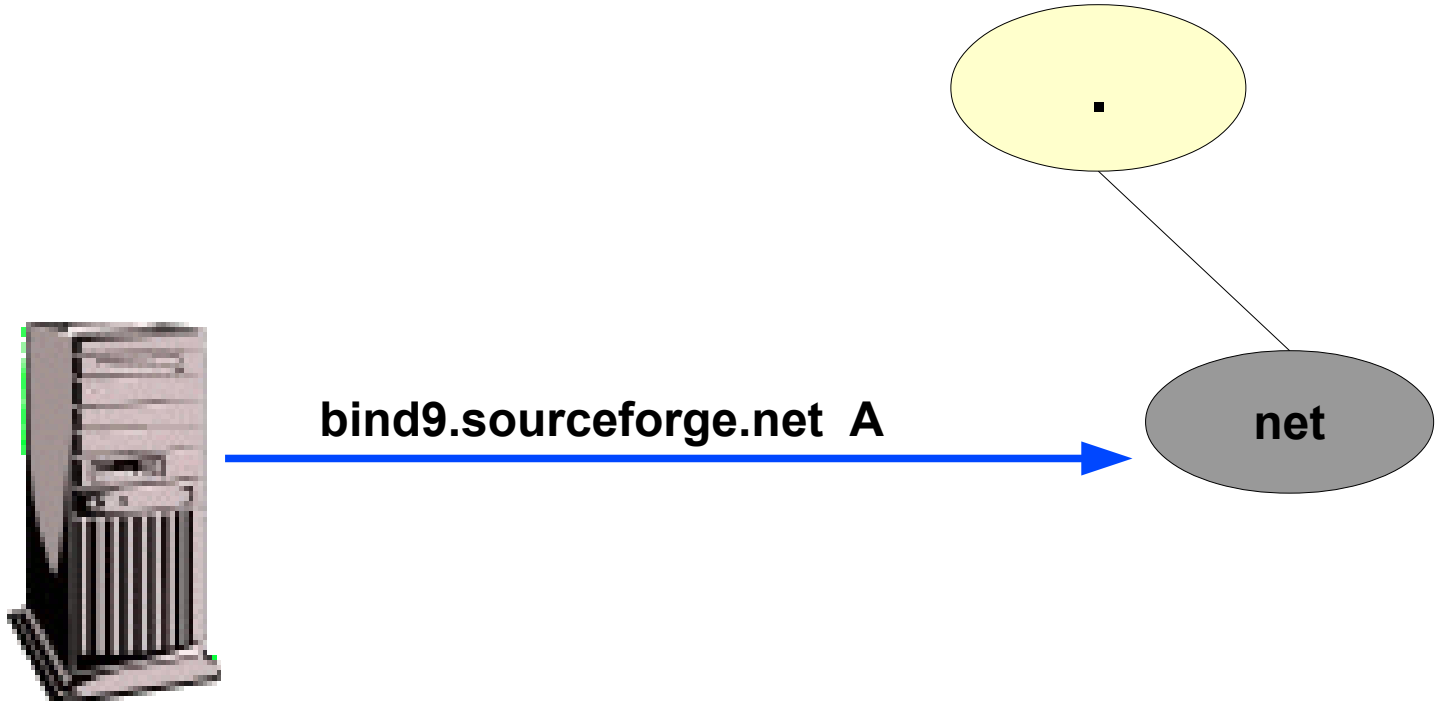
DNS trabaja bajo el principio de delegación

El servidor del dominio raiz responde diceindo que le pregunte al servidor del dominio .net



DNS trabaja bajo el principio de delegación

El servidor local entonces le pregunta al servidor del dominio net

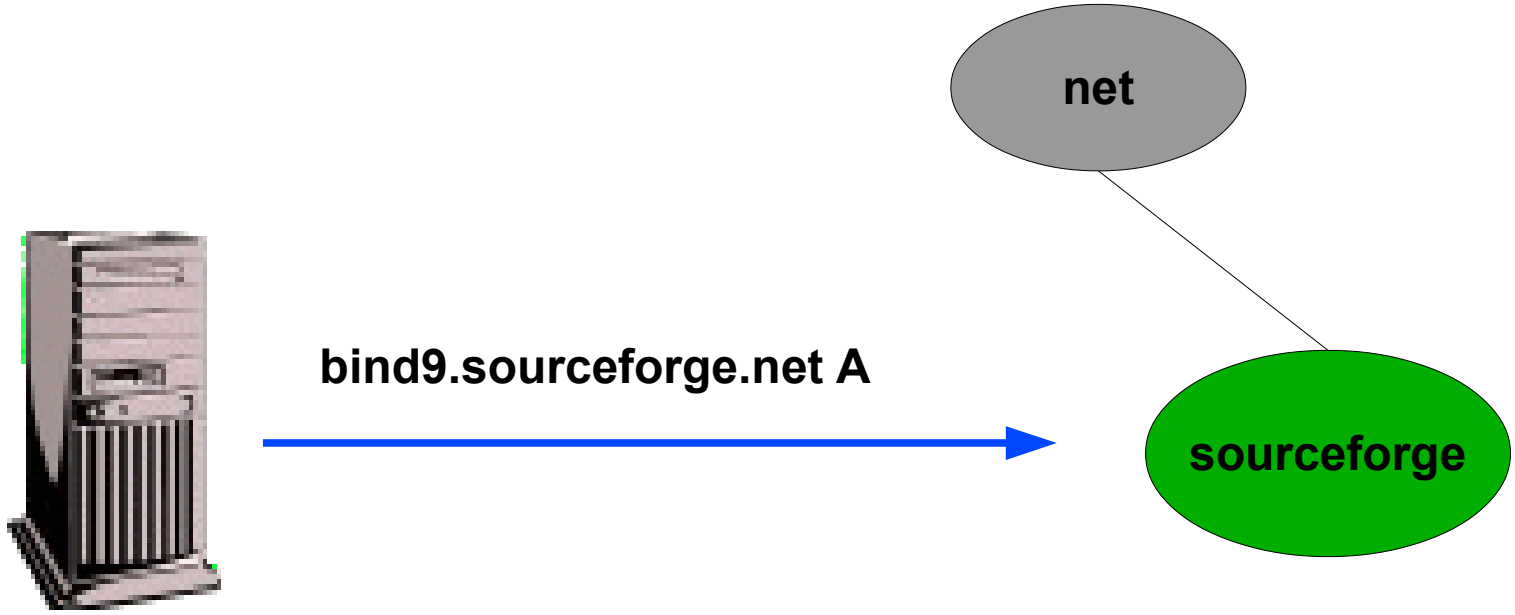


DNS trabaja bajo el principio de delegación

Éste a su vez responde que le pregunte al servidor del dominio sourceforge.net

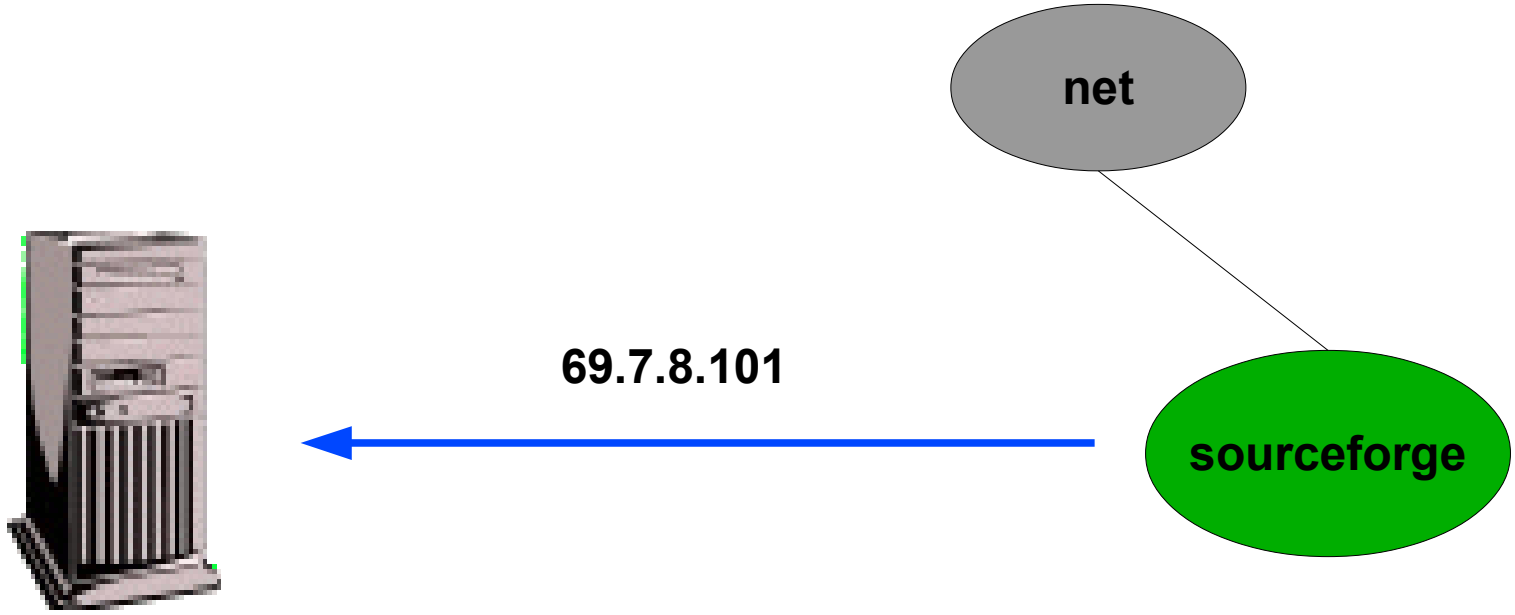
DNS trabaja bajo el principio de delegación

El servidor local le pregunta al servidor del dominio sourceforge.net



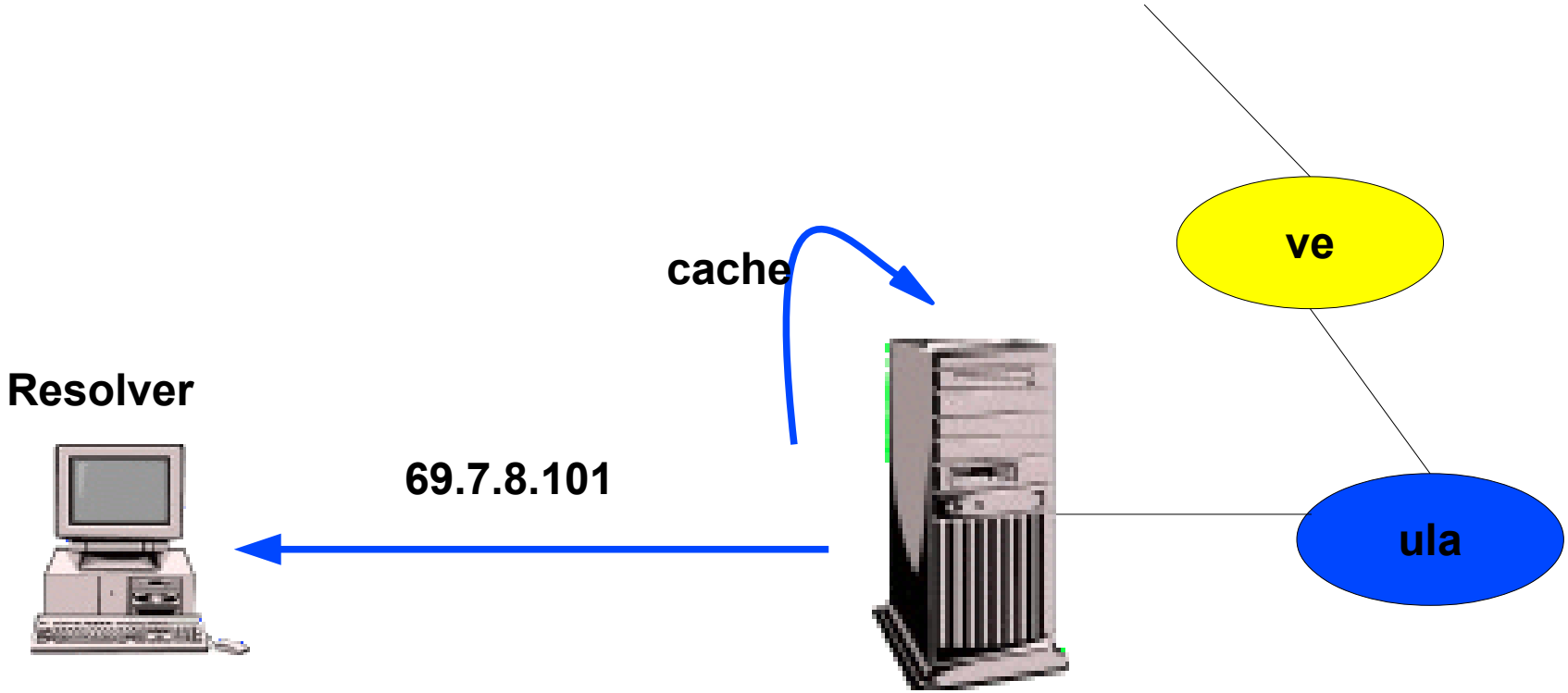
DNS trabaja bajo el principio de delegación

El servidor del dominio sourceforge.net responde finalmente con la dirección solicitada



DNS trabaja bajo el principio de delegación

El servidor local le responde al cliente la solicitud y almacena la dirección en cuestión en el cache para responder a futuras solicitudes



DNS fue introducido en 1984

Desde sus inicios el desarrollo de DNS se llevó a cabo sin considerar la seguridad pues el tamaño de la red, el tipo de usuarios, y la utilización de recursos no propiciaban incidentes de seguridad.

A medida que Internet creció y más usuarios se conectaron aparecieron amenazas en los diferenctes servicios y se hizo necesario considerar la seguridad de los mismos.

Vulnerabilidades en DNS

- DNS utiliza el protocolo UDP el cual acarrea múltiples vulnerabilidades.

- UDP no tiene mecanismos para verificar el origen de cada paquete. Esto lo hace susceptible a spoofing y ataques de origen

Vulnerabilidades en DNS

- Muchas de ellas son de índole general, como las mencionadas anteriormente, pero hay unas cuantas que son intrínsecas al protocolo DNS como tal.

Vulnerabilidades en DNS

Los cuatro problemas principales que podemos encontrar en DNS son:

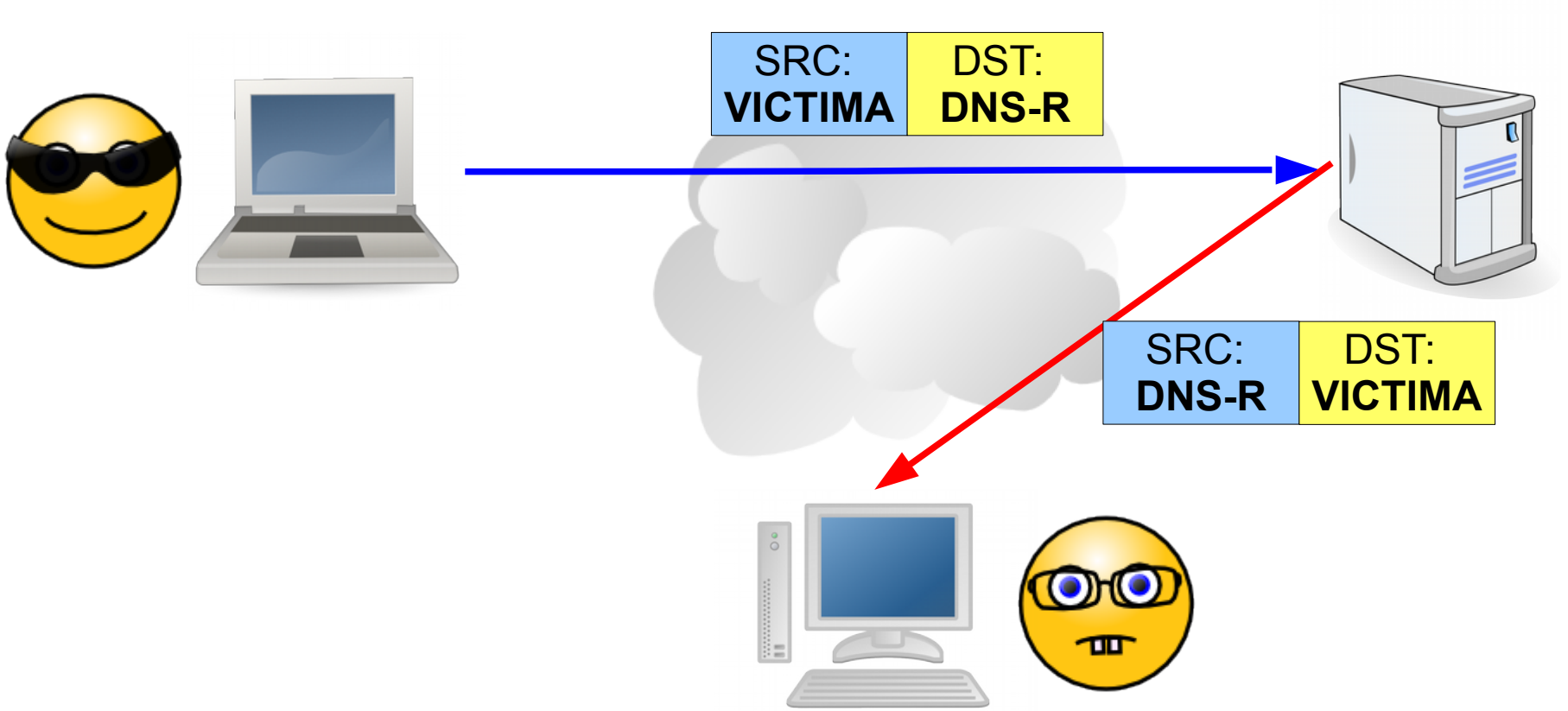
- cache spoofing
- traffic diversion
- distributed denial-of-service attacks (DDoS)
- buffer overruns.

Ataque de Negación de Servicio Distribuido (DDoS) utilizando el DNS

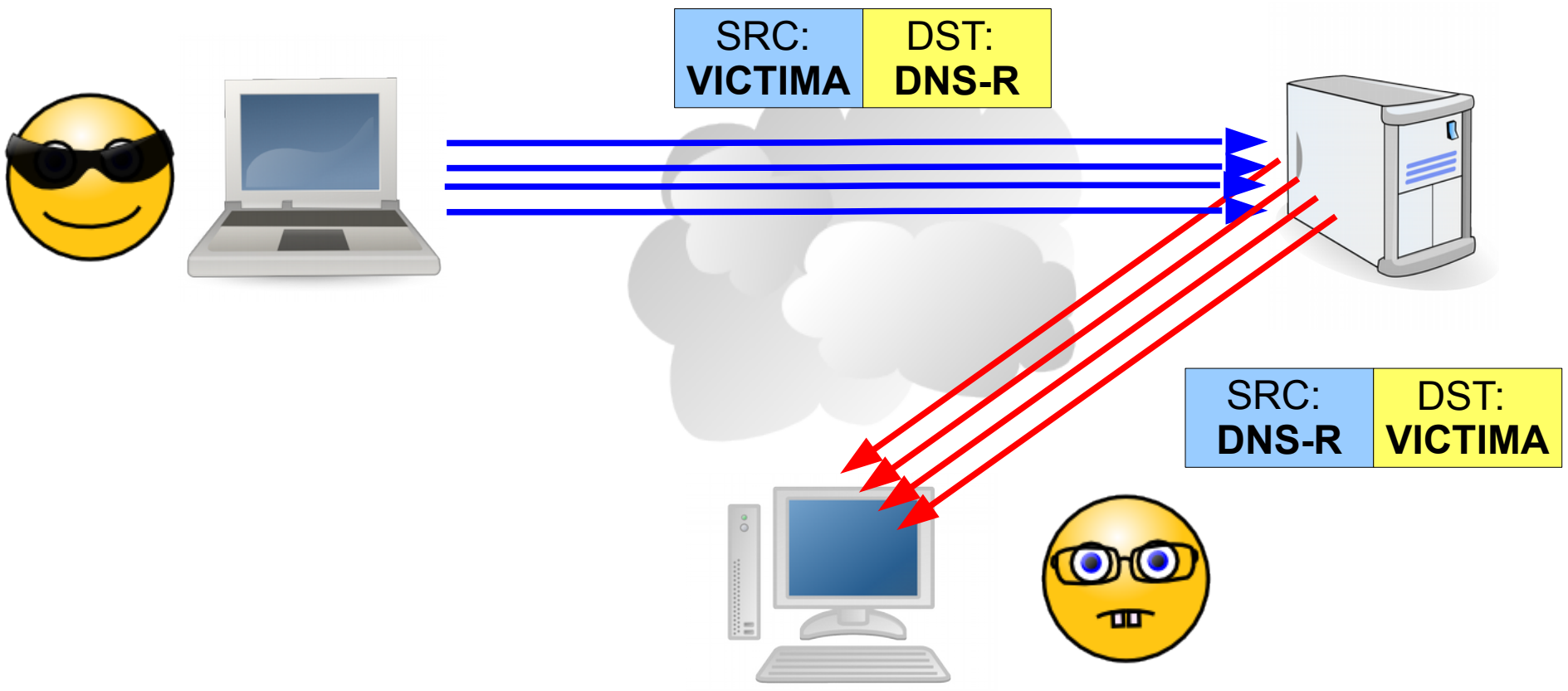
Ataque de Negación de Servicio Distribuido usando el DNS

Ingredientes

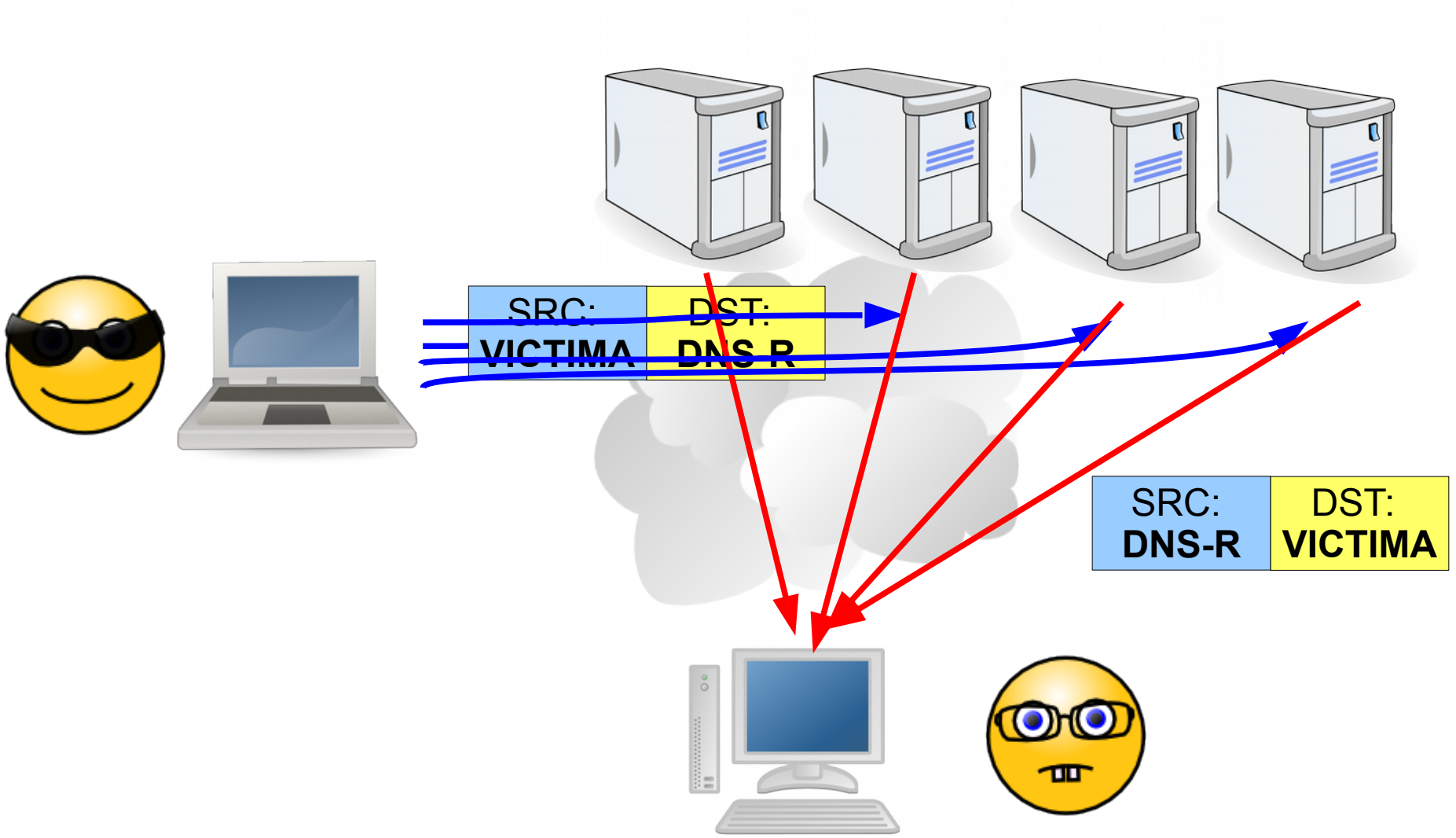
- Atacante
- Víctima
- Servidores de DNS con recursión activada



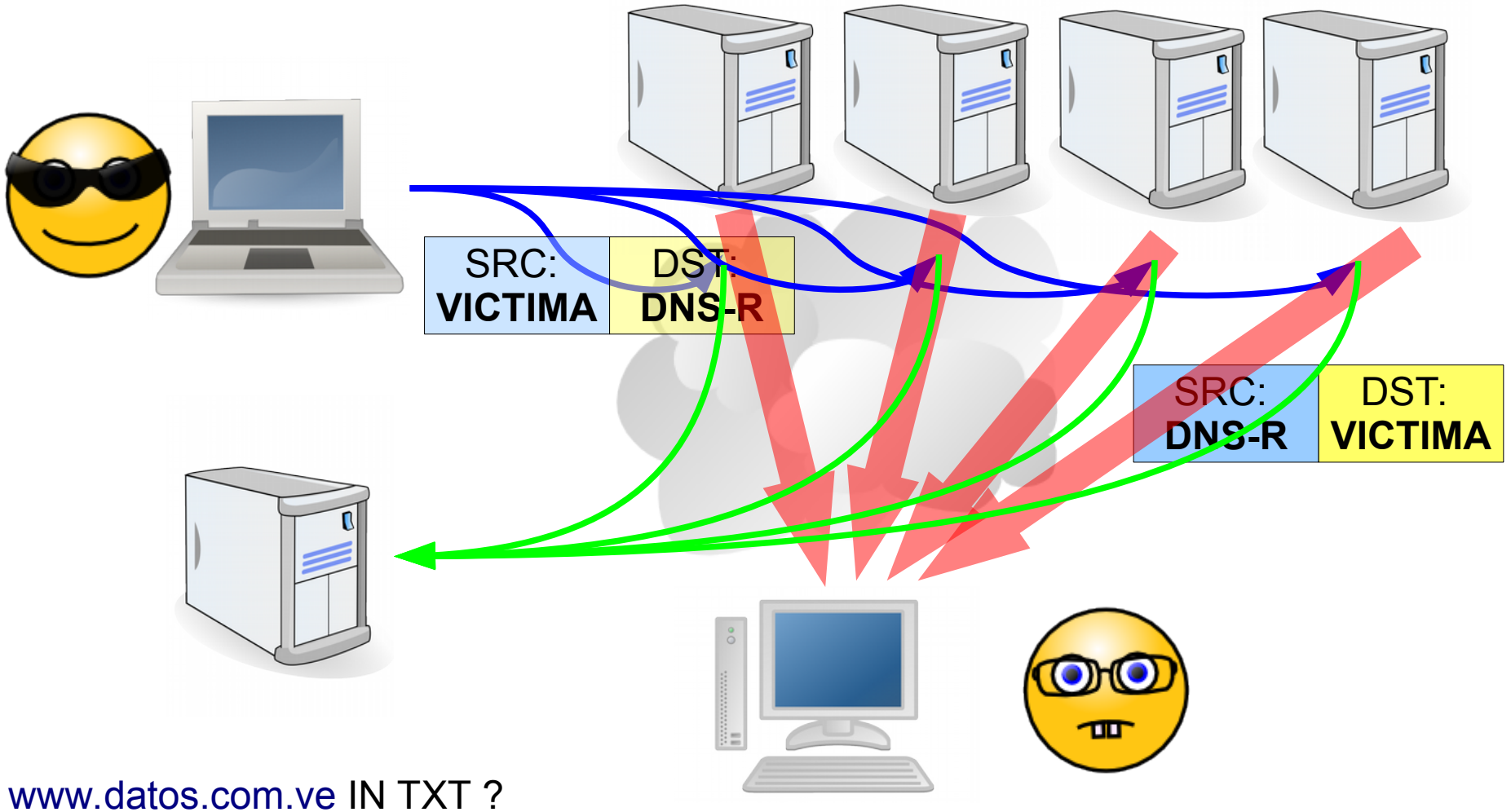
Ataque de Negación de Servicio Distribuido usando el DNS



Ataque de Negación de Servicio Distribuido usando el DNS



Ataque de Negación de Servicio Distribuido usando el DNS



www.datos.com.ve IN TXT ?

XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXX

512 Bytes (en UDP) o más (con TCP)

Ataque de Negación de Servicio Distribuido usando el DNS (Soluciones)

- **Contribuir NO teniendo servidores de DNS con recursión abierta al público**
  - En ISPs
  - Universidades
  - En servidores de prueba, laboratorios, etc.

- **Validar los orígenes a nivel de enrutadores: (ej: cisco)**

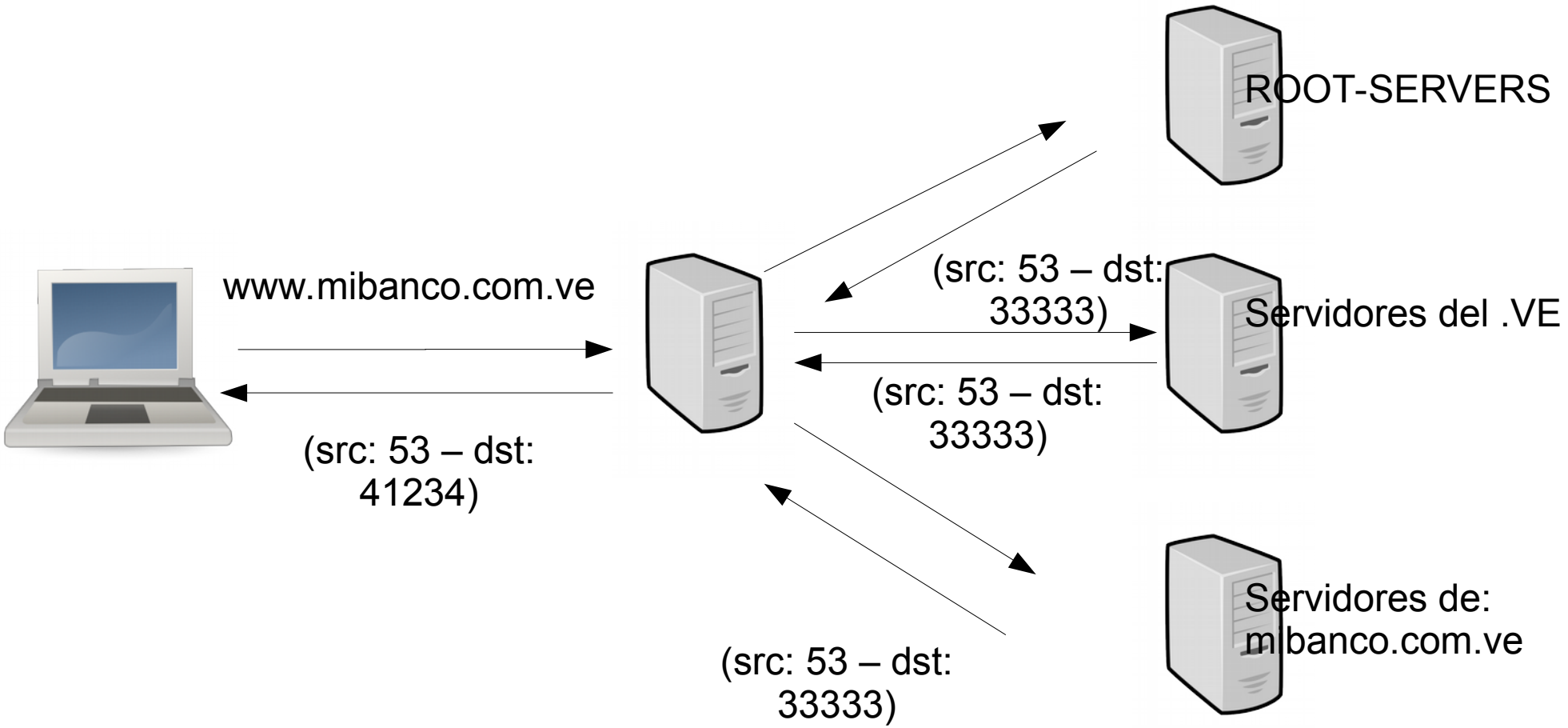
Ataque de “Cumpleaños” (envenenamiento de cache)

Ataque de “Cumpleaños” (Envenenamiento de Cache)

Cual es la probabilidad que en un grupo de 23 personas, 2 o mas tengan la misma fecha de nacimiento?

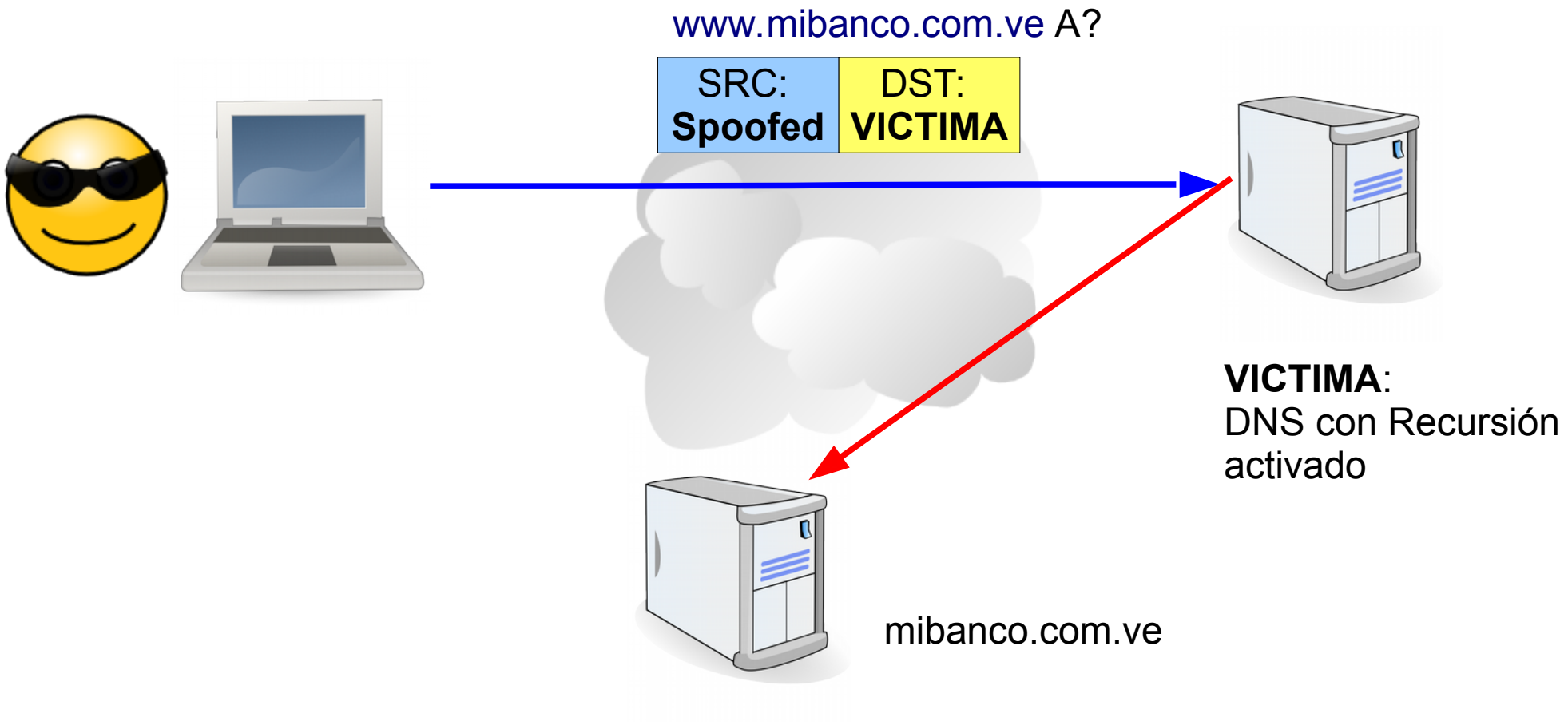
Probabilidad es mayor a ½ !!!

Ataque de “Cumpleaños” (Envenenamiento de Cache)

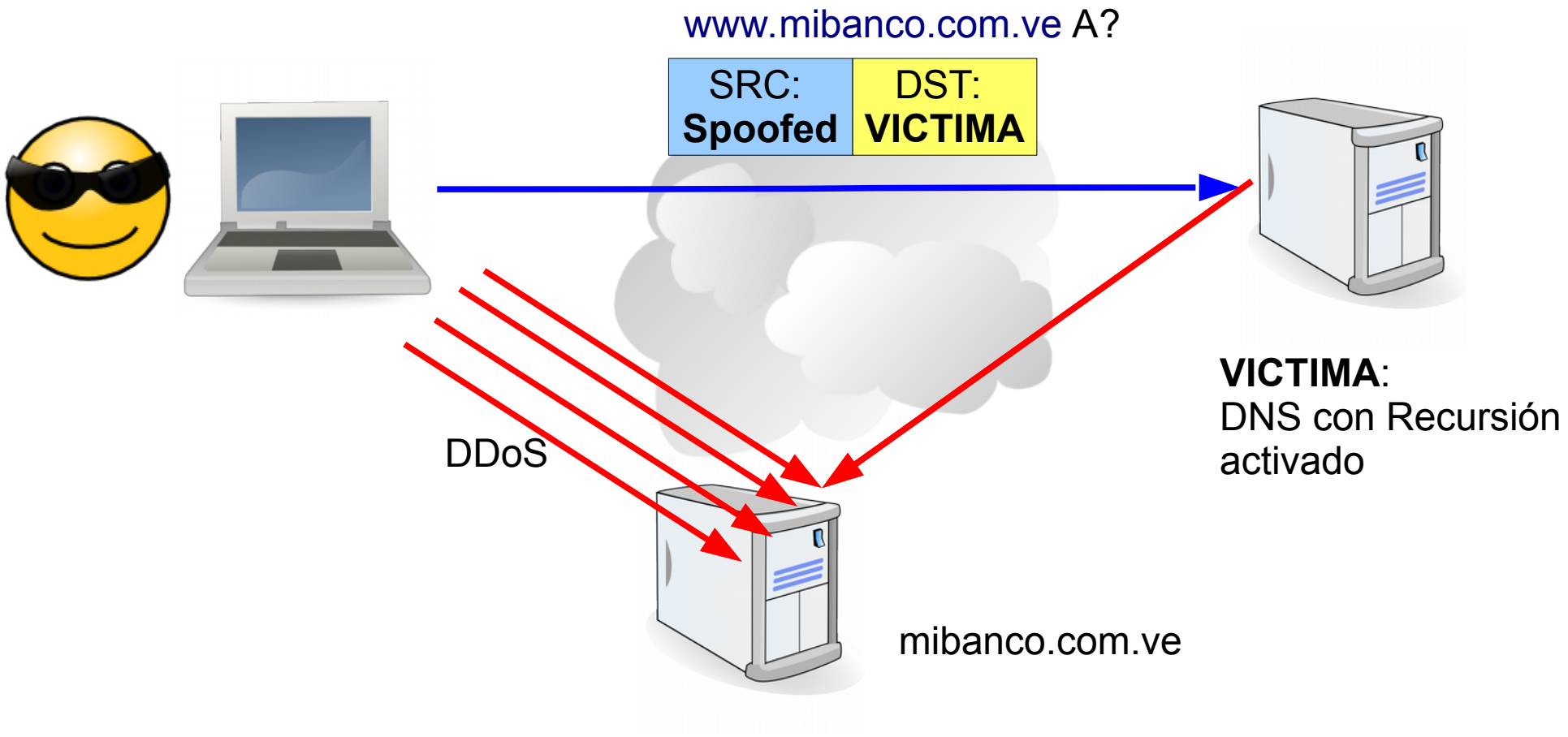


SRC: SR SPT: 33333	DST: S DPT:53

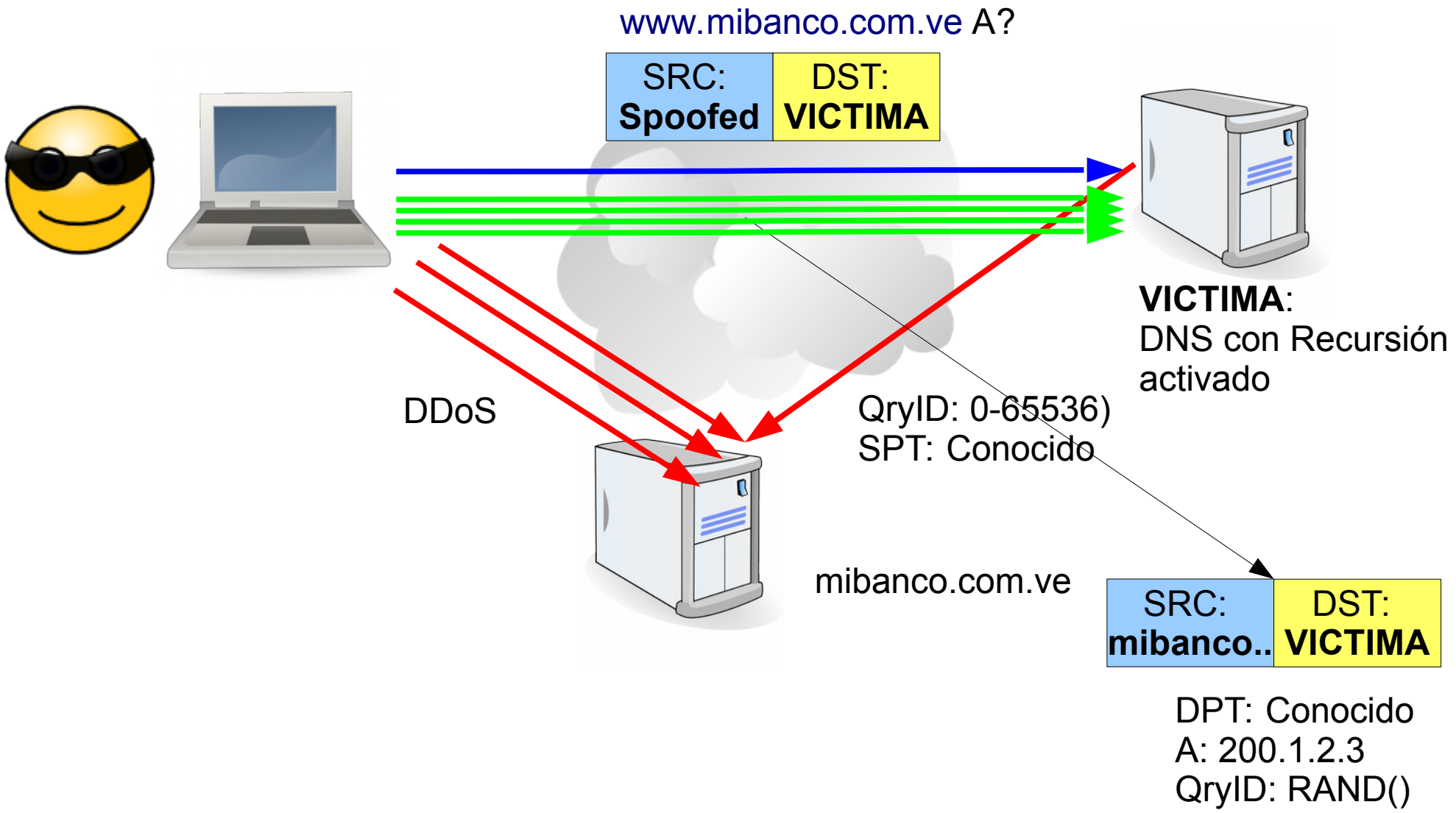
Ataque de “Cumpleaños” (Envenenamiento de Cache)



Ataque de “Cumpleaños” (Envenenamiento de Cache)



Ataque de “Cumpleaños” (Envenenamiento de Cache)



Ataque de “Cumpleaños” (Envenenamiento de Cache)

- Probabilidad de éxito de 100% con alrededor de 700 paquetes

- Básicamente es una carrera entre el servidor autoritativo y el atacante, donde el atacante ya tiene la ventaja

Ataque de “Cumpleaños” (Envenenamiento de Cache)

Para minimizar la probabilidad de envenenamiento:

- Actualizar a BIND9
- Utilizar servidores de DNS que generen puertos origen aleatorios
- Mejorar el generador de números aleatorios (/dev/random, algún dispositivo externo,etc.)
- Las mismas previsiones del ataque DDoS

DNSSEC

Las extensiones de seguridad de DNS solucionan los problemas relacionados con:

- Integridad de los datos
- Source spoofing

¡No! protegen contra

- distributed denial-of-service attacks (DDoS)
- buffer overruns.

DNSSEC

Esta extensión utiliza tecnología ya conocida (PKI) para funcionar y es totalmente transparente para los usuarios, es decir, ellos no necesitan realizar ninguna acción para beneficiarse con DNSSEC.

DNSSEC

- Esta extensión utiliza tecnología ya conocida (PKI) para funcionar

- Es totalmente transparente para los usuarios, es decir, ellos no necesitan realizar ninguna acción para beneficiarse con DNSSEC.

- Esta disponible sólo a partir de Bind9. ¡**Actualice**!

DNSSEC

- El objetivo principal de esta extensión es garantizar la autenticidad e integridad de los datos
- Esto se logra utilizando criptografía de clave asimétrica y simétrica
- Puede utilizar tanto RSA como DSA
- Se debe utilizar en conjunto con la restricciones para transferencias de zonas

Configuración DNSSEC

- Lo primero que debemos hacer es **restringir** la transferencias de zonas.

- El comportamiento predefinido *(no es por defecto porque no esta dañado)* de DNS es aceptar desde cualquier máquina una lista completa de sus registros

Configuración DNSSEC

```
options {
        allow-transfer { 192.168.4.154; };
};
```

O de una forma más específica

```
type master;
zone "ejemplo.com" {
        file "db.ejemplo.com";
        allow-transfer { 192.168.4.154; };
};
```

Configuración DNSSEC

- Para asegurar mucho más nuestro servidor restringimos también las actualizaciones

```
zone "ejemplo.com" {
	type master;
	file "db.ejemplo.com";
	update-policy {
		grant allowed-www-updater self
www.ejemplo.com A;
	};
};
```

Configuración DNSSEC

- Finalmente configuramos TSIG para asegurar la autenticidad de la fuente de los datos

En el Master

```
key tsig-signing. {
        algorithm hmac-md5;
        secret "mZiMNOUYQPMNwsDzrX2ENw==";
};
zone "ejemplo.com" {
        type master;
        file "db.ejeplo.com";
        allow-transfer { key tsig-signing; };
};
```

Configuración DNSSEC

- En el Slave

```
key tsig-signing. {
        algorithm hmac-md5;
        secret "mZiMNOUYQPMNwsDzrX2ENw==";
};
server 192.168.4.47 {
        transfer-format many-answers;
        keys { tsig-signing.; };
};
zone "ejemplo.com" {
        type slave;
        file "bak.ejemplo.com";
        allow-transfer { none; };
};
```

Finalmente para proteger el servidor contra cache-poisoning

- Evitamos la recursión

```
options {
        recursion no;
};
```