# Domain Name System

Thomas Espitau

Cachan Réseau à Normale Sup'

10 février 2014

# Un petit test...

- A quel site correspond l'adresse : `138.231.136.67` ?
- `www.crans.org`
- Et quelle est l'(une des) adresses de www.google.com ?
- `173.194.45.50`
- Comment faire lien entre les deux ? C'est le but de ce séminaire

# Un petit test...

- A quel site correspond l'adresse : `138.231.136.67` ?
  - `www.crans.org`
- Et quelle est l'(une des) adresses de www.google.com ?
  - `173.194.45.50`
- Comment faire lien entre les deux ? C'est le but de ce séminaire

# Un petit test...

- A quel site correspond l'adresse : `138.231.136.67` ?
  - `www.crans.org`
- Et quelle est l'(une des) adresses de www.google.com ?
  - `173.194.45.50`
- Comment faire lien entre les deux ? C'est le but de ce séminaire

# Un petit test...

- A quel site correspond l'adresse : `138.231.136.67` ?
  - `www.crans.org`
- Et quelle est l'(une des) adresses de www.google.com ?
  - `173.194.45.50`
- Comment faire lien entre les deux ? C'est le but de ce séminaire

# Un petit test...

- A quel site correspond l'adresse : `138.231.136.67` ?
  - `www.crans.org`
- Et quelle est l'(une des) adresses de www.google.com ?
  - `173.194.45.50`
- Comment faire lien entre les deux ? C'est le but de ce séminaire

# Un petit test...

- A quel site correspond l'adresse : `138.231.136.67` ?
  - `www.crans.org`
- Et quelle est l'(une des) adresses de www.google.com ?
  - `173.194.45.50`
- Comment faire lien entre les deux ? C'est le but de ce séminaire

# Sommaire



# Architecture centralisée

- Système hierarchisé à structure arborescente :
- Une racine unique (`.`, distribuée sur plusieurs serveurs)
- Des domaines de haut niveau (TLD, Top Level Domains distribués par l'IANA (`http://www.iana.org/domains/root/db/`) - *Internet Assigned Numbers Authority* -
  - génériques (gTLD) : `org` , `com`, ...
  - country code (ccTLD) : `fr`, `tv`, ...
- Chaque TLD distribue des domaines (`crans.org.`, ...)

# Architecture centralisée

- Système hierarchisé à structure arborescente :
- Une racine unique ( . , distribuée sur plusieurs serveurs)
- Des domaines de haut niveau (TLD, Top Level Domains distribués par l'IANA (`http://www.iana.org/domains/root/db/`) - *Internet Assigned Numbers Authority* -
  - génériques (gTLD) : `org` , `com`, ...
  - country code (ccTLD) : `fr`, `tv`, ...
- Chaque TLD distribue des domaines (`crans.org.`, ...)

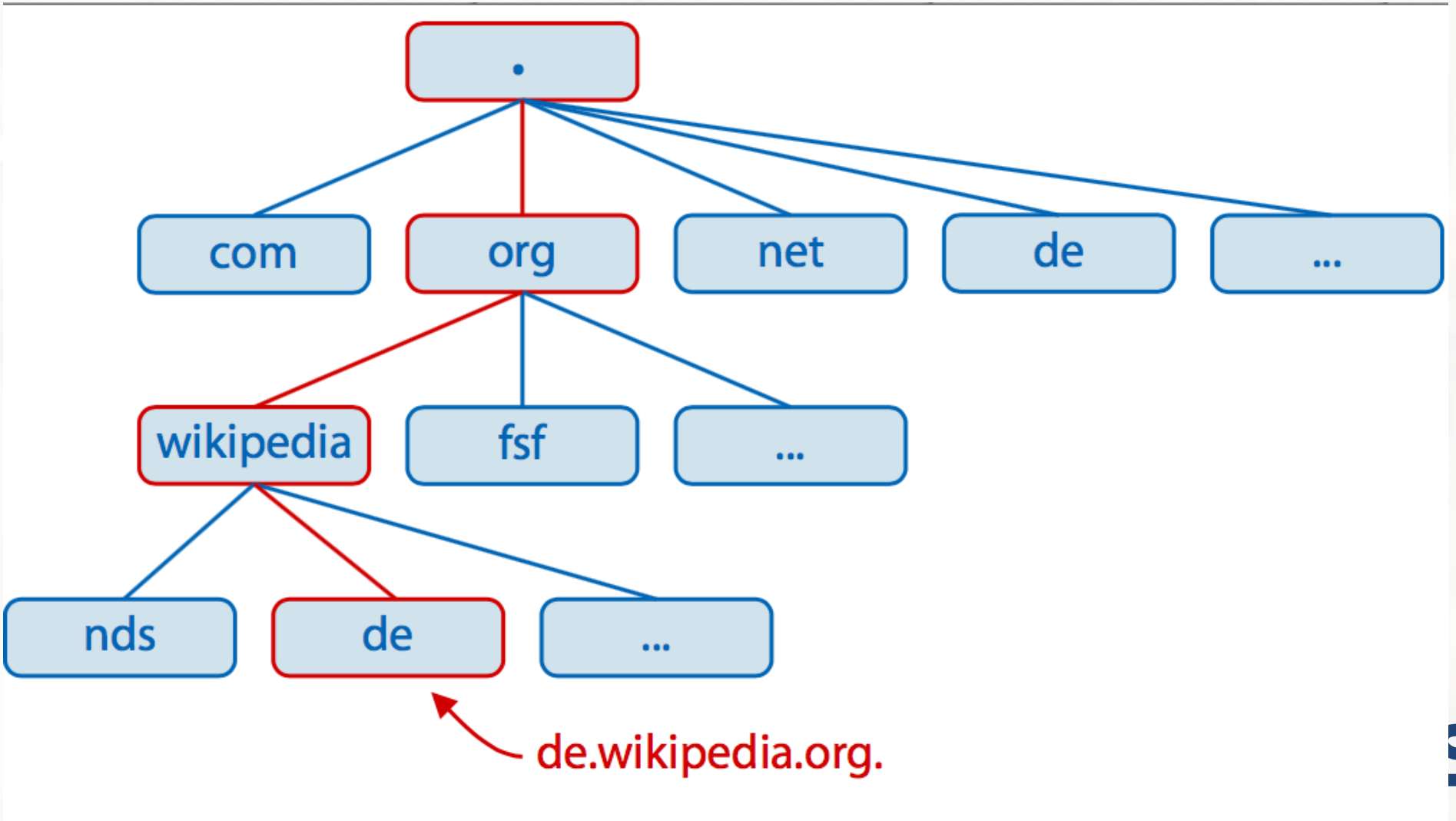
# Architecture centralisée

- Système hierarchisé à structure arborescente :
- Une racine unique ( . , distribuée sur plusieurs serveurs)
- Des domaines de haut niveau (TLD, Top Level Domains distribués par l'IANA (`http://www.iana.org/domains/root/db/`) - *Internet Assigned Numbers Authority* -
  - génériques (gTLD) : `org` , `com`, ...
  - country code (ccTLD) : `fr`, `tv`, ...
- Chaque TLD distribue des domaines (`crans.org`, ...)

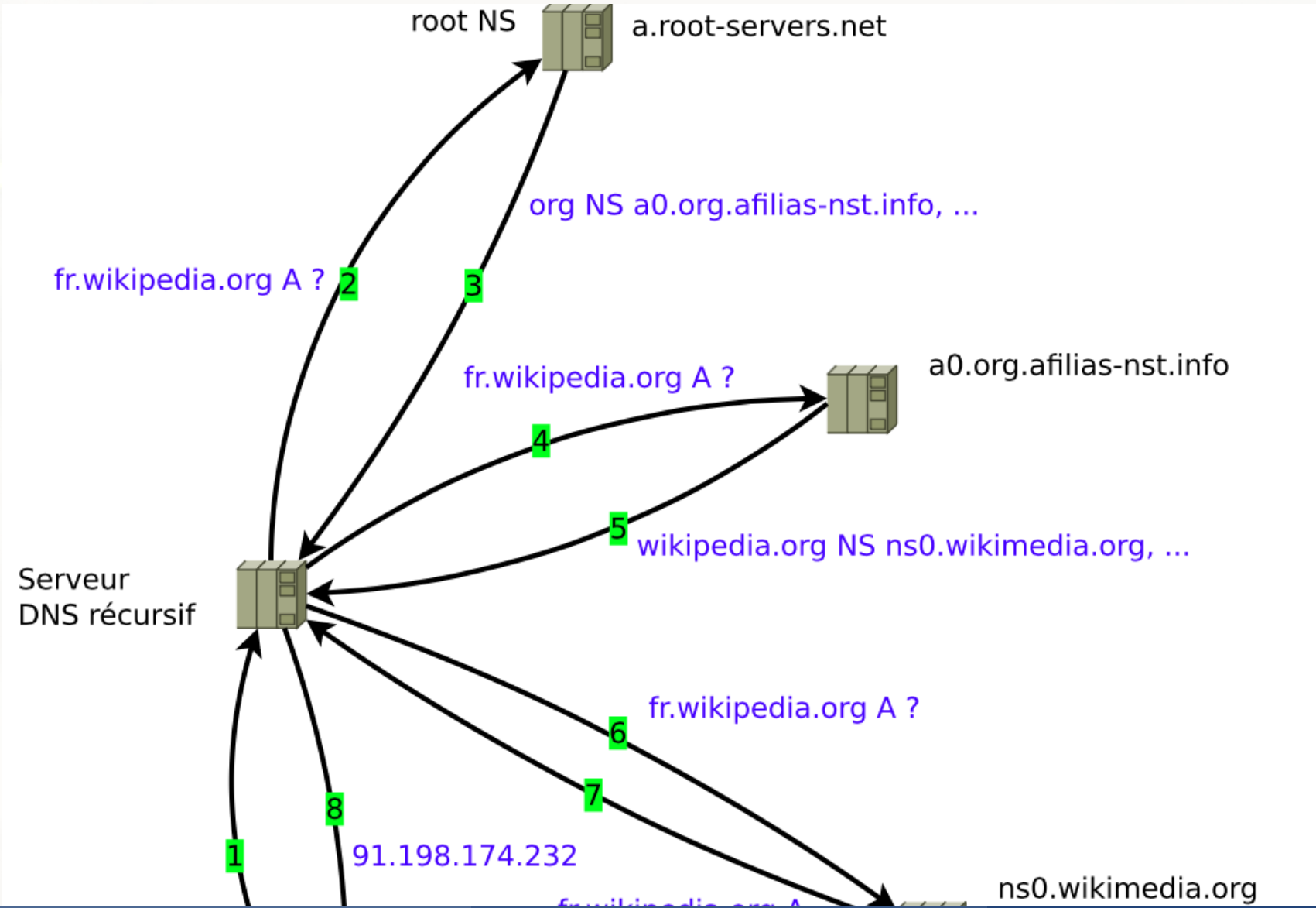
# Architecture centralisée

- Système hierarchisé à structure arborescente :
- Une racine unique ( . , distribuée sur plusieurs serveurs)
- Des domaines de haut niveau (TLD, Top Level Domains distribués par l'IANA (`http://www.iana.org/domains/root/db/`) - *Internet Assigned Numbers Authority* -
  - génériques (gTLD) : `org` , `com`, ...
  - country code (ccTLD) : `fr`, `tv`, ...
- Chaque TLD distribue des domaines (`crans.org.`, ...)

# Achitecture du DNS



# Exemple de résolution



# Résolution inverse

- Un nom correspond à une adresse (`IN A zamok.crans.org.` → `138.231.136.1`)
- Mais il existe des zones inverses (`in-addr.arpa.` pour l'IPv4, `ip6.arpa.` pour l'IPv6).
- Mapping inverse : `IN PTR 1.136.231.138.in-addr.arpa.` → `zamok.crans.org.`
- .arpa est hérité du passage DARPA -> ARPANET où les machines d'arpanet ont été converties en nom de domaines en .arpa.

# Résolution inverse

- Un nom correspond à une adresse (`IN A zamok.crans.org.` → `138.231.136.1`)
- Mais il existe des zones inverses (`in-addr.arpa.` pour l'IPv4, `ip6.arpa.` pour l'IPv6).
- Mapping inverse : `IN PTR 1.136.231.138.in-addr.arpa.` → `zamok.crans.org.`
- .arpa est hérité du passage DARPA -> ARPANET où les machines d'arpanet ont été converties en nom de domaines en .arpa.

# Résolution inverse

- Un nom correspond à une adresse (`IN A zamok.crans.org.` → `138.231.136.1`)
- Mais il existe des zones inverses (`in-addr.arpa.` pour l'IPv4, `ip6.arpa.` pour l'IPv6).
- Mapping inverse : `IN PTR 1.136.231.138.in-addr.arpa.` → `zamok.crans.org.`
- .arpa est hérité du passage DARPA -> ARPANET où les machines d'arpanet ont été converties en nom de domaines en .arpa.

# Deux types de serveurs de noms

- *Autoritaires* : distribuent les entrées de zones données dont ils sont « propriétaires »
  - `ovh.crans.org`
  - `sable.crans.org`
  - `freebox.crans.org`
- *Récursif* : permet la résolution de noms pour les machines clientes
  - `/etc/resolv.conf`

# Serveurs racines

- Serveur racine du DNS : serveur DNS qui répond aux requêtes qui concernent les noms de domaine de premier niveau (top-level domain, TLD)
- redirige vers le serveur DNS de premier niveau concerné.
- 13 racines du Domain Name System d'Internet gérées sous l'autorité de l'ICANN. 9 sont distribués en suivant *anycast*.
- `x.root-servers.net` (`x` allant de `a` à `m`).

# Quelques subtilités

- Serveurs secondaires : robustesse
- Caching :
  - Sauvegarde réponse de résolution de nom pendant un laps de temps (*time to live* (TTL)) – généralement une journée environ – rapidité, mais parfois pas à jour

# Quelques subtilités

- Serveurs secondaires : robustesse
- Caching :
  - Sauvegarde réponse de résolution de nom pendant un laps de temps (*time to live* (TTL)) – généralement une journée environ – rapidité, mais parfois pas à jour

# Quelques subtilités

- Serveurs secondaires : robustesse
- Caching :
  - Sauvegarde réponse de résolution de nom pendant un laps de temps (*time to live* (TTL)) – généralement une journée environ –

rapidité , mais parfois pas à jour

# Quelques subtilités

- Serveurs secondaires : robustesse
- Caching :
  - Sauvegarde réponse de résolution de nom pendant un laps de temps (*time to live* (TTL)) – généralement une journée environ –

rapidité , mais parfois pas à jour

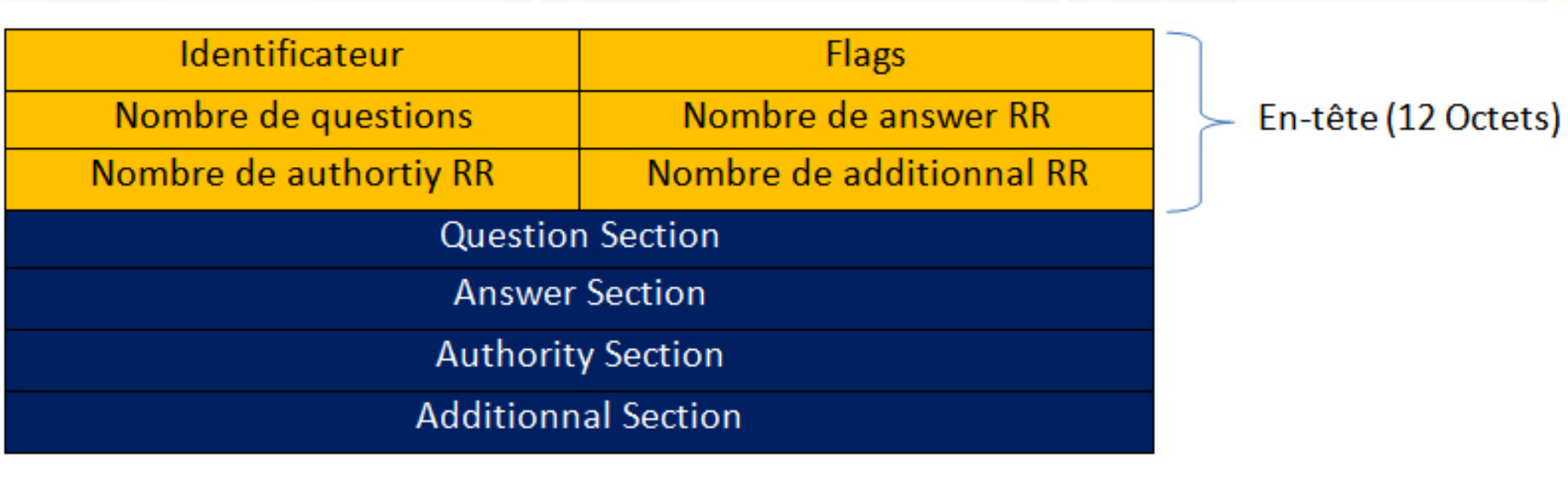
# Quelques subtilités

- Serveurs secondaires : robustesse
- Caching :
  - Sauvegarde réponse de résolution de nom pendant un laps de temps (*time to live* (TTL)) – généralement une journée environ –

rapidité , mais parfois pas à jour

# Sommaire




Identificateur
Flags
Nombre de questions
Nombre de answer RR
Nombre de authortiy RR
Nombre de additionnal RR
En-tête (12 Octets)
Question Section
Answer Section
Authority Section
Additionnal Section

# En tête de message

- **Identificateur** : Nombre entier représentant une requête qui doit être recopié lors de la réponse permettant à l'application de départ de pouvoir identifier le datagramme de retour.(16 bits)
- **Nombre de questions** : Spécifie le nombre d'entrée dans la section " Question " (16 bits)
- **Nombre de answer RR** : Spécifie le nombre d'entrée dans la section " Answer " (16 bits)
- **Nombre de authority RR** : Spécifie le nombre d'entrée dans la section " Authority " (16 bits)
- **Nombre de additionnal RR** : Spécifie le nombre d'entrée dans la section " Additionnal " (16 bits)

# En tête de message

- **Identificateur** : Nombre entier représentant une requête qui doit être recopié lors de la réponse permettant à l'application de départ de pouvoir identifier le datagramme de retour.(16 bits)
- **Nombre de questions** : Spécifie le nombre d'entrée dans la section " Question " (16 bits)
- **Nombre de answer RR** : Spécifie le nombre d'entrée dans la section " Answer " (16 bits)
- **Nombre de authority RR** : Spécifie le nombre d'entrée dans la section " Authority " (16 bits)
- **Nombre de additionnal RR** : Spécifie le nombre d'entrée dans la section " Additionnal " (16 bits)

# En tête de message

- **Identificateur** : Nombre entier représentant une requête qui doit être recopié lors de la réponse permettant à l'application de départ de pouvoir identifier le datagramme de retour.(16 bits)
- **Nombre de questions** : Spécifie le nombre d'entrée dans la section " Question " (16 bits)
- **Nombre de answer RR** : Spécifie le nombre d'entrée dans la section " Answer " (16 bits)
- **Nombre de authority RR** : Spécifie le nombre d'entrée dans la section " Authority " (16 bits)
- **Nombre de additionnal RR** : Spécifie le nombre d'entrée dans la section " Additionnal " (16 bits)

# En tête de message

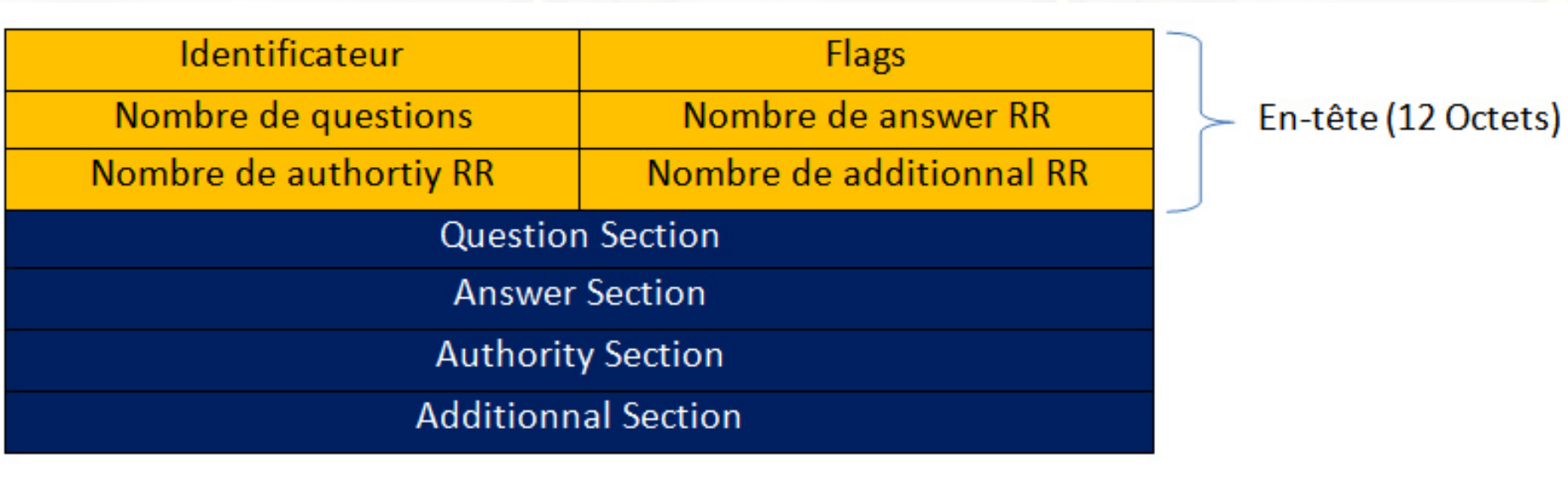
- **Identificateur** : Nombre entier représentant une requête qui doit être recopié lors de la réponse permettant à l'application de départ de pouvoir identifier le datagramme de retour.(16 bits)
- **Nombre de questions** : Spécifie le nombre d'entrée dans la section " Question " (16 bits)
- **Nombre de answer RR** : Spécifie le nombre d'entrée dans la section " Answer " (16 bits)
- **Nombre de authority RR** : Spécifie le nombre d'entrée dans la section " Authority " (16 bits)
- **Nombre de additionnal RR** : Spécifie le nombre d'entrée dans la section " Additionnal " (16 bits)

# En tête de message

- **Identificateur** : Nombre entier représentant une requête qui doit être recopié lors de la réponse permettant à l'application de départ de pouvoir identifier le datagramme de retour.(16 bits)
- **Nombre de questions** : Spécifie le nombre d'entrée dans la section " Question " (16 bits)
- **Nombre de answer RR** : Spécifie le nombre d'entrée dans la section " Answer " (16 bits)
- **Nombre de authority RR** : Spécifie le nombre d'entrée dans la section " Authority " (16 bits)
- **Nombre de additionnal RR** : Spécifie le nombre d'entrée dans la section " Additionnal " (16 bits)

# En tête de message

- **Identificateur** : Nombre entier représentant une requête qui doit être recopié lors de la réponse permettant à l'application de départ de pouvoir identifier le datagramme de retour.(16 bits)
- **Nombre de questions** : Spécifie le nombre d'entrée dans la section " Question " (16 bits)
- **Nombre de answer RR** : Spécifie le nombre d'entrée dans la section " Answer " (16 bits)
- **Nombre de authority RR** : Spécifie le nombre d'entrée dans la section " Authority " (16 bits)
- **Nombre de additionnal RR** : Spécifie le nombre d'entrée dans la section " Additionnal " (16 bits)


Identificateur
Flags
Nombre de questions
Nombre de answer RR
Nombre de authortiy RR
Nombre de additionnal RR
En-tête (12 Octets)
Question Section
Answer Section
Authority Section
Additionnal Section

# Flags

- **Qr** : Indique s'il s'agit d'une requête (0) ou d'une réponse (1). (1 bit),
- **Opcode** : Spécifie le type de requête (4 bits)
  - 0 : Requête standard (Query)
  - 1 : Requête inverse (IQuery)
  - 2 : Statut du serveur (Status)
  - 3-15 : Réservés pour utilisation future

# Flags

- **Aa** : ( *" Authoritative Answer "*) indique si le serveur de nom répondant à la requête est autoritaire sur le domaine (1 bit).
- **Tc** : Indique que ce message a été tronqué. ( 0 lorsque le protocole TCP est utilisé, si UDP est utilisé ce flag peut être positionné à 1 si la réponse excède 512 octets )(1bit)
- **Rd** : Permet de demander la récursivité en le mettant à 1 (1 bit)
- **Ra** : Indique que la récursivité est autorisée (1 bit).
- **Z** : Réservé à utilisation future. Il doit être positionné à 0 (1 bit)
- **Rcode** : Indique le type de réponse (4 bits) :
  - 0 : Pas d'erreur
  - 1 : Erreur de format dans la requête
  - 2 : Problème sur serveur
  - 3 : Le nom n'existe pas
  - 4 : Non implémenté
  - 5 : Refus
  - 6-15 : Réservés.

# Les petites briques... les Ressources Records

- **Nom** : Nom du domaine où se trouve le RR.
- **Classe** : Identifie une famille de protocoles ou une instance d'un protocole, parmis :
  - `In` Internet
  - `Cs` Class Csnet (obsolete)
  - `Ch` Chaos (obsolete (pour les Lisp-Machines au MIT en ... 1975)
  - `Hs` Hesiod (plus implémenté)
- **Longueur** Indique la longueur des données suivantes.(16bit)

- **Nom** : Nom du domaine où se trouve le RR.
- **Classe** : Identifie une famille de protocoles ou une instance d'un protocole, parmis :
  - `In` Internet
  - `Cs` Class Csnet (obsolete)
  - `Ch` Chaos (obsolete (pour les Lisp-Machines au MIT en ... 1975)
  - `Hs` Hesiod (plus implémenté)
- **Longueur** Indique la longueur des données suivantes.(16bit)

- **Nom** : Nom du domaine où se trouve le RR.
- **Classe** : Identifie une famille de protocoles ou une instance d'un protocole, parmis :
  - `In` Internet
  - `Cs` Class Csnet (obsolete)
  - `Ch` Chaos (obsolete (pour les Lisp-Machines au MIT en ... 1975)
  - `Hs` Hesiod (plus implémenté)
- **Longueur** Indique la longueur des données suivantes.(16bit)

- **Nom** : Nom du domaine où se trouve le RR.
- **Classe** : Identifie une famille de protocoles ou une instance d'un protocole, parmis :
  - `In` Internet
  - `Cs` Class Csnet (obsolete)
  - `Ch` Chaos (obsolete (pour les Lisp-Machines au MIT en ... 1975)
  - `Hs` Hesiod (plus implémenté)
- **Longueur** Indique la longueur des données suivantes.(16bit)

# Types

- `SOA` : Informations sur la zone
- `NS` : Serveur de noms correspondant à la zone
- `MX` : Serveur d'échange d'emails
- `A` : Adresse IP
- `AAAA` : Adresse IPv6
- `CNAME` : Redirection de nom de domaine
- `PTR` : Correspondance inverse

# `dig` session !

- **D**omain **I**nformation **G**roper

# Sommaire



# Principe des attaques sur DNS

- Leurer les serveurs DNS pour forcer une réponse/ forcer l'écriture dans le cache :
  - l'attaquant peut lier une adresse donée avec une IP choisie.
  - Permet de rediriger vers un site donné.

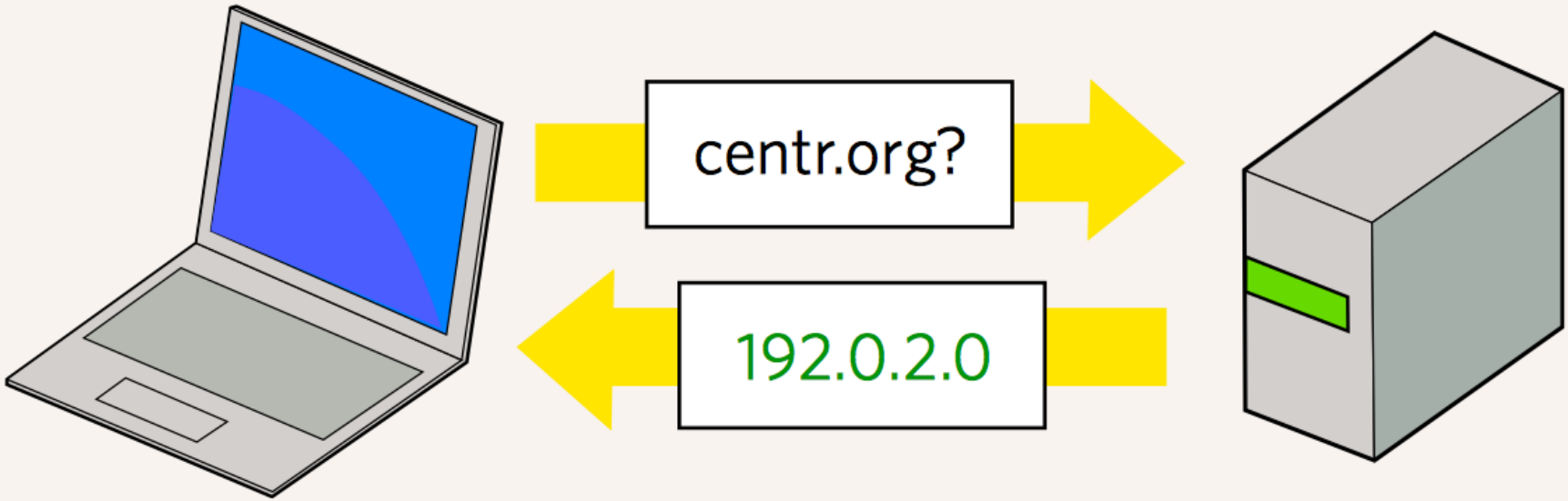
# Principe des attaques sur DNS

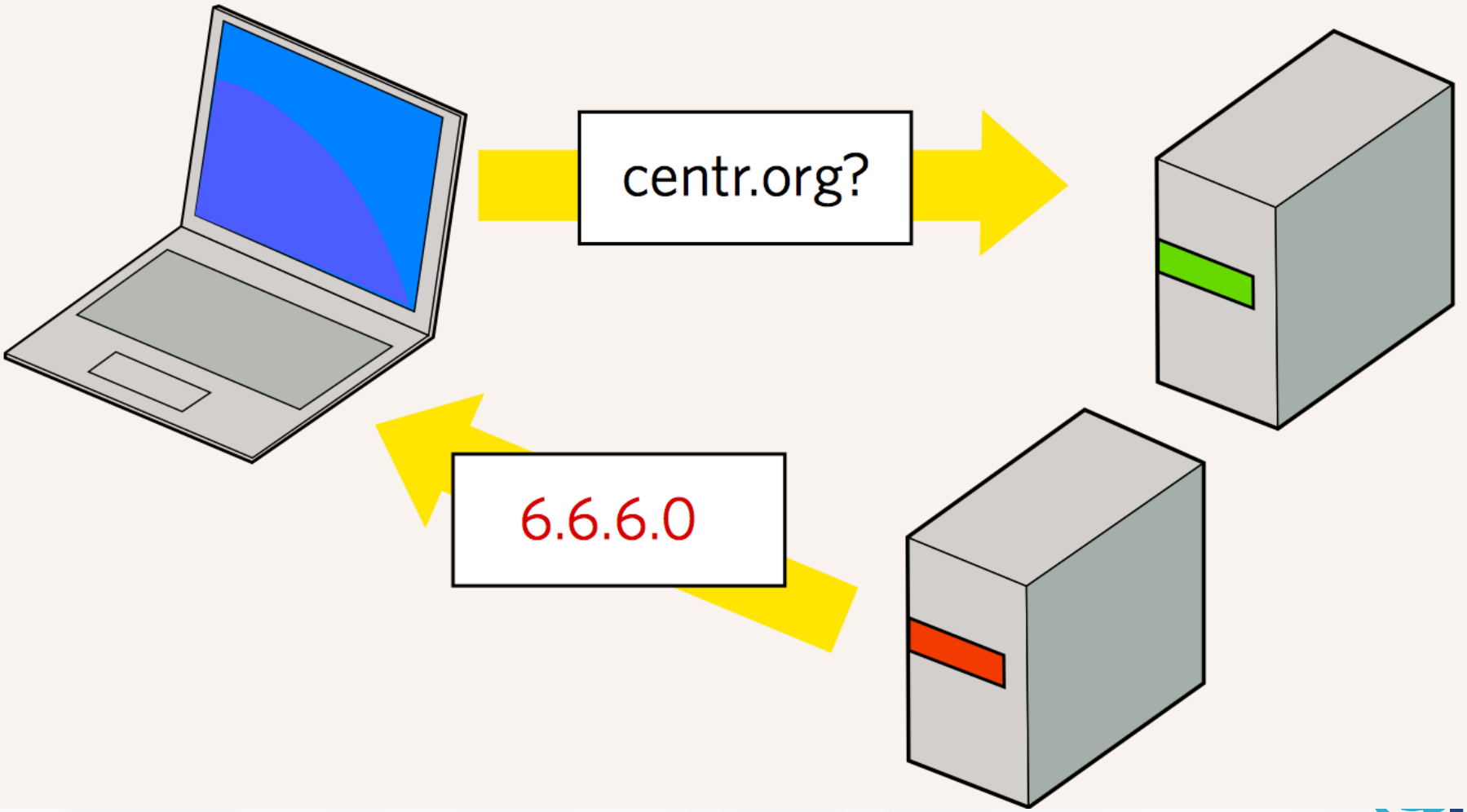
- Leurer les serveurs DNS pour forcer une réponse/ forcer l'écriture dans le cache :
- l'attaquant peut lier une adresse donée avec une IP choisie.
- Permet de rediriger vers un site donné.

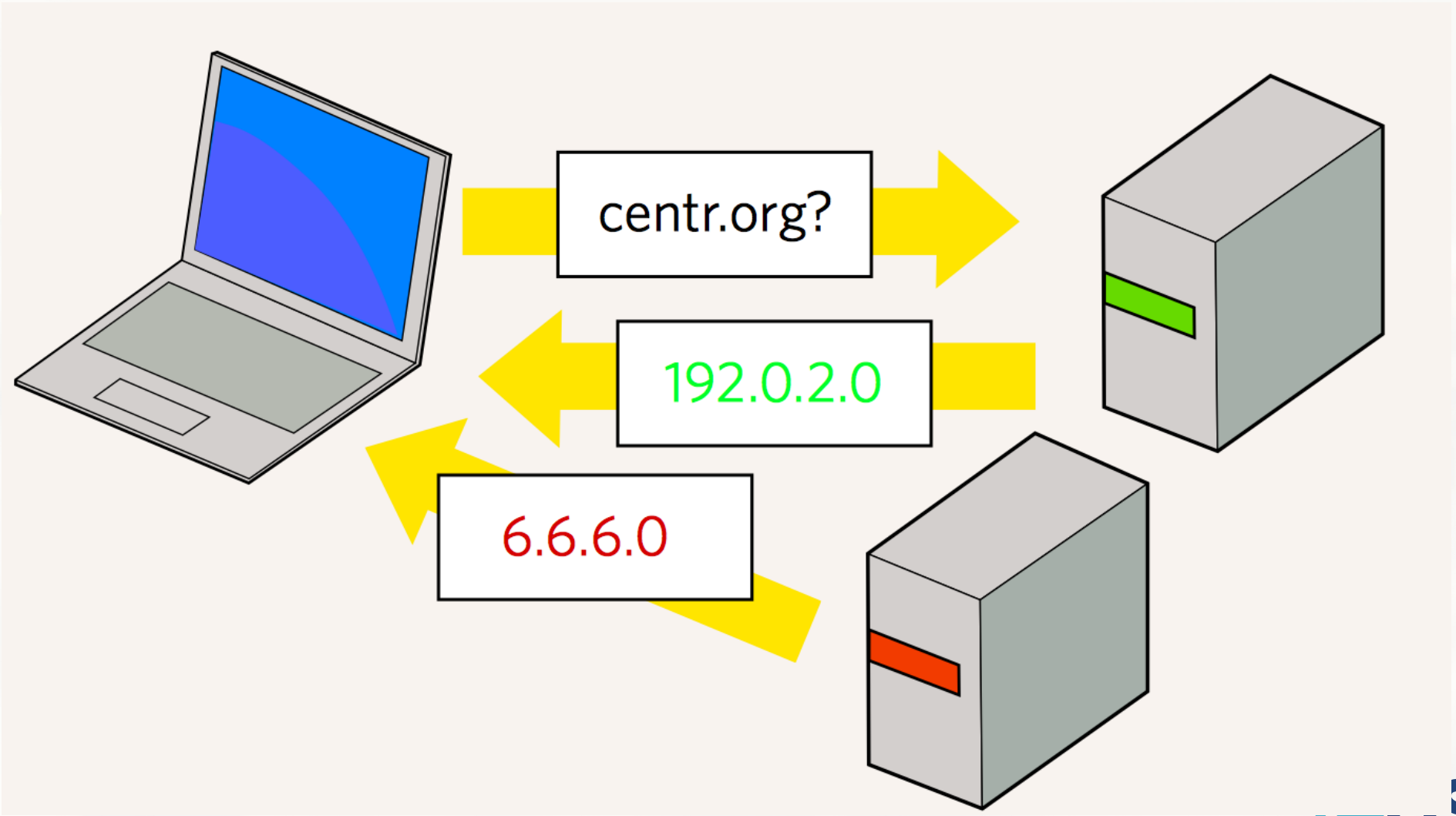
# Principe des attaques sur DNS

- Leurer les serveurs DNS pour forcer une réponse/ forcer l'écriture dans le cache :
- l'attaquant peut lier une adresse donée avec une IP choisie.
- Permet de rediriger vers un site donné.

# Prinicpe(2)




centr.org?
6.6.6.0


centr.org?
192.0.2.0
6.6.6.0

# Comment ?

- Par accès direct au serveur
- Par le paradoxe des anniversaires :
  - Alice envoie un grand nombre de requêtes au serveur cible A, en spécifiant le nom de domaine X dont elle voudrait obtenir l'IP
  - En même temps, elle se fait passer pour un autre serveur de nom B en préparant des réponses qui y ressemblent, mais avec des identifiants de transaction aléatoires de manière à produire un paquet comme celui attendu par le serveur cible. Elle spécifie par ailleurs un nouvel IP w.x.y.z
  - si le paquet correspond à ce qui était attendu par le serveur cible, alors en l'absence d'une protection particulière, insère l'information malveillante dans le cache
  - Bob demande à accéder à X, l'IP qu'il reçoit n'est plus celle de X mais bien w.x.y.z
- Permet de rediriger *massivement* vers un site donné.

# Comment ?

- Par accès direct au serveur
- Par le paradoxe des anniversaires :
  - Alice envoie un grand nombre de requêtes au serveur cible A, en spécifiant le nom de domaine X dont elle voudrait obtenir l'IP
  - En même temps, elle se fait passer pour un autre serveur de nom B en préparant des réponses qui y ressemblent, mais avec des identifiants de transaction aléatoires de manière à produire un paquet comme celui attendu par le serveur cible. Elle spécifie par ailleurs un nouvel IP w.x.y.z
  - si le paquet correspond à ce qui était attendu par le serveur cible, alors en l'absence d'une protection particulière, insère l'information malveillante dans le cache
  - Bob demande à accéder à X, l'IP qu'il reçoit n'est plus celle de X mais bien w.x.y.z
- Permet de rediriger *massivement* vers un site donné.

# DNSSEC

- Patch contre les attaques courantes pas une solution à long terme
- **D**omain **N**ame **S**ystem **S**ecurity **E**xtensions
- Ne protège pas les canaux mais les records : reste efficace même si un serveur intermédiaire tombe
- DNSSEC signe cryptographiquement les records : le client peut récupérer la signature et vérifier l'exactitude des données (s'il possède la clef publique du serveur)

# DNSSEC

- Patch contre les attaques courantes pas une solution à long terme
- **D**omain **N**ame **S**ystem **S**ecurity **E**xtensions
- Ne protège pas les cannaux mais les records : reste efficace même si un serveur intermediaire tombe
- DNSSEC signe cryptographiquement les records : le client peut récuperer la signature et vérifier l'exactidude des données (s'il possède la clef publique du serveur)

# DNSSEC

- Patch contre les attaques courantes pas une solution à long terme
- **D**omain **N**ame **S**ystem **S**ecurity **E**xtensions
- Ne protège pas les cannaux mais les records : reste efficace même si un serveur intermédiaire tombe.
- DNSSEC signe cryptographiquement les records : le client peut récuperer la signature et vérifier l'exactidude des données (s'il possède la clef publique du serveur)

# DNSSEC

- Patch contre les attaques courantes pas une solution à long terme
- **D**omain **N**ame **S**ystem **S**ecurity **E**xtensions
- Ne protège pas les cannaux mais les records : reste efficace même si un serveur intermédiaire tombe.
- DNSSEC signe cryptographiquement les records : le client peut récuperer la signature et vérifier l'exactidude des données (s'il possède la clef publique du serveur)

# DNSSEC

- Patch contre les attaques courantes pas une solution à long terme
- **D**omain **N**ame **S**ystem **S**ecurity **E**xtensions
- Ne protège pas les cannaux mais les records : reste efficace même si un serveur intermédiaire tombe.
- DNSSEC signe cryptographiquement les records : le client peut récuperer la signature et vérifier l'exactidude des données (s'il possède la clef publique du serveur)

# DNSSEC

- Patch contre les attaques courantes pas une solution à long terme
- **D**omain **N**ame **S**ystem **S**ecurity **E**xtensions
- Ne protège pas les cannaux mais les records : reste efficace même si un serveur intermédiaire tombe.
- DNSSEC signe cryptographiquement les records : le client peut récuperer la signature et vérifier l'exactidude des données (s'il possède la clef publique du serveur)

# DNSSEC(2)

- Possibilité de déléguer les signatures : les TLD peuvent annoncer qu'un sous domaine est signé.
- Création d'un chaine de confiance

# DNSSEC(2)

- Possibilité de déléguer les signatures : les TLD peuvent annoncer qu'un sous domaine est signé.
- Création d'un chaine de confiance

# Une faille dans DNSSEC

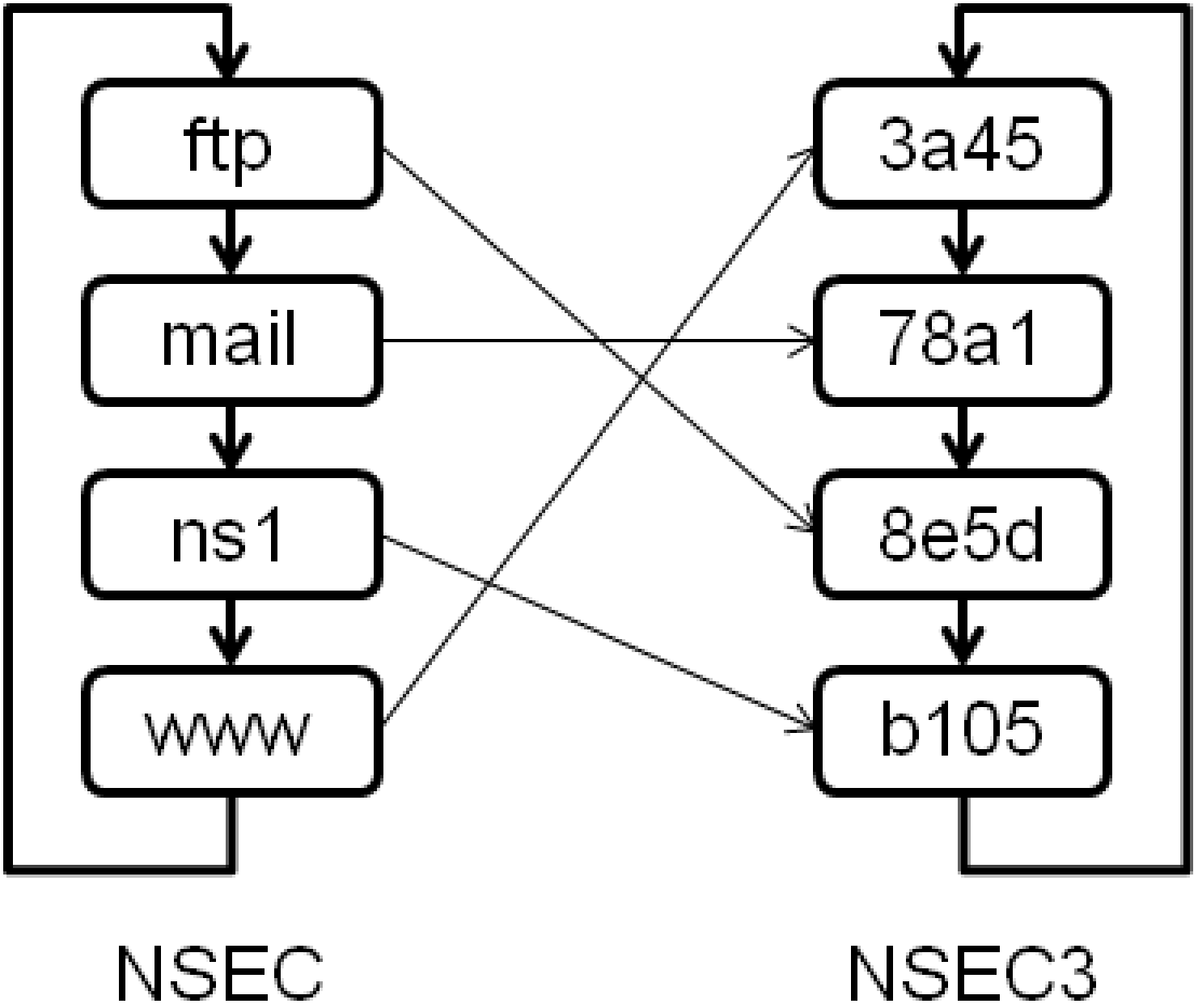
- En cas de requête d'un domaine n'existant pas, le serveur renvoie une réponse **NXDOMAIN**
- Dans DNSSEC, le serveur signe la non existence en renvoyant les noms de domaines contigus dans l'ordre lexicographique.
- Solution : Hasher les noms avant de les trier

# Une faille dans DNSSEC

- En cas de requête d'un domaine n'existant pas, le serveur renvoie une réponse **NXDOMAIN**
- Dans DNSSEC, le serveur signe la non existence en renvoyant les noms de domaines contigus dans l'ordre lexicographique.
- Solution : Hasher les noms avant de les trier

# Une faille dans DNSSEC

- En cas de requête d'un domaine n'existant pas, le serveur renvoie une réponse **NXDOMAIN**
- Dans DNSSEC, le serveur signe la non existence en renvoyant les noms de domaines contigus dans l'ordre lexicographique.
- Solution : Hasher les noms avant de les trier


ftp
mail
ns1
www
3a45
78a1
8e5d
b105
NSEC
NSEC3