Domain Name Security Extentions

Eastlake and Kaufman

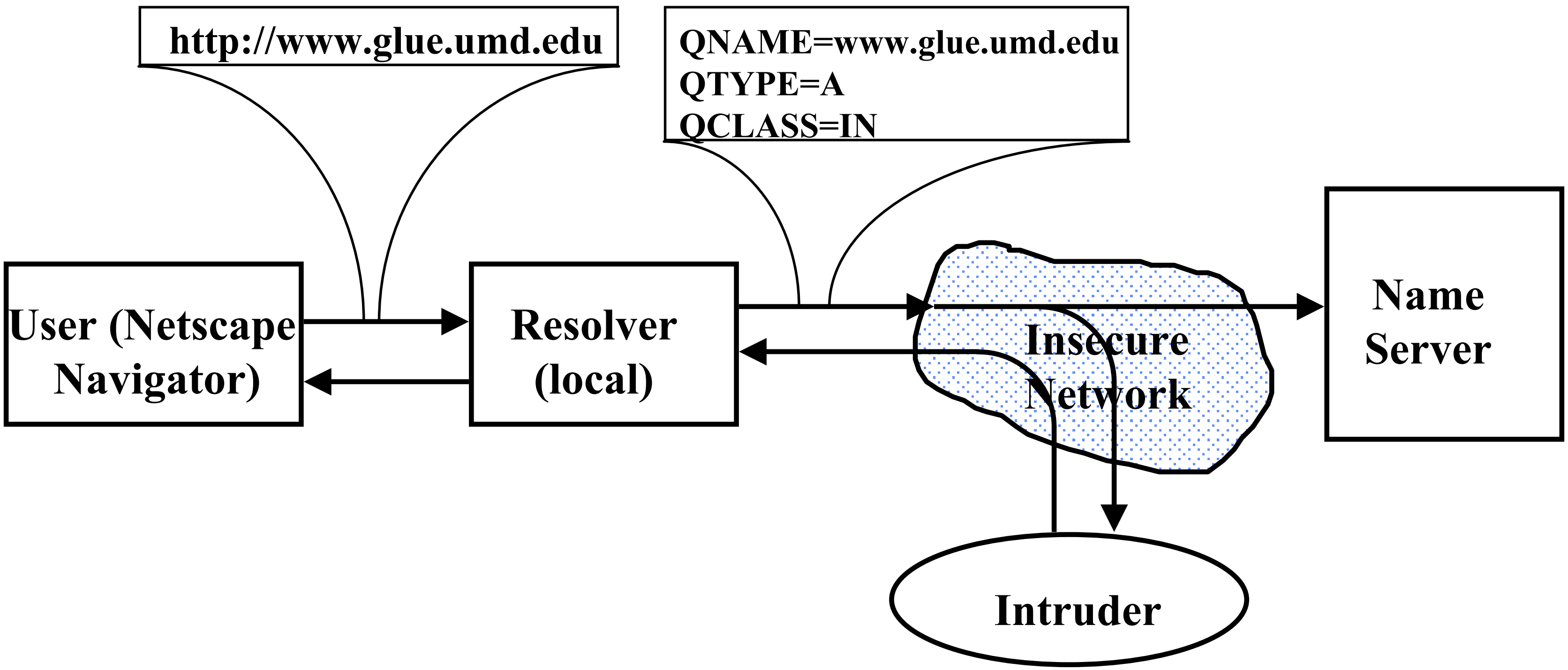
November 1996

Domain Name System Security Extensions

- The DNS :
  - lacks mechanisms to ensure data integrity and authentication
  - doesn’t care about secrecy

- Goals of the security extentions => provide for :
  - data integrity and response authentication through use of digital signatures
  - query authentication (optional)
  - security even through non-security-aware servers
  - provide for storage of authenticated public keys in the DNS

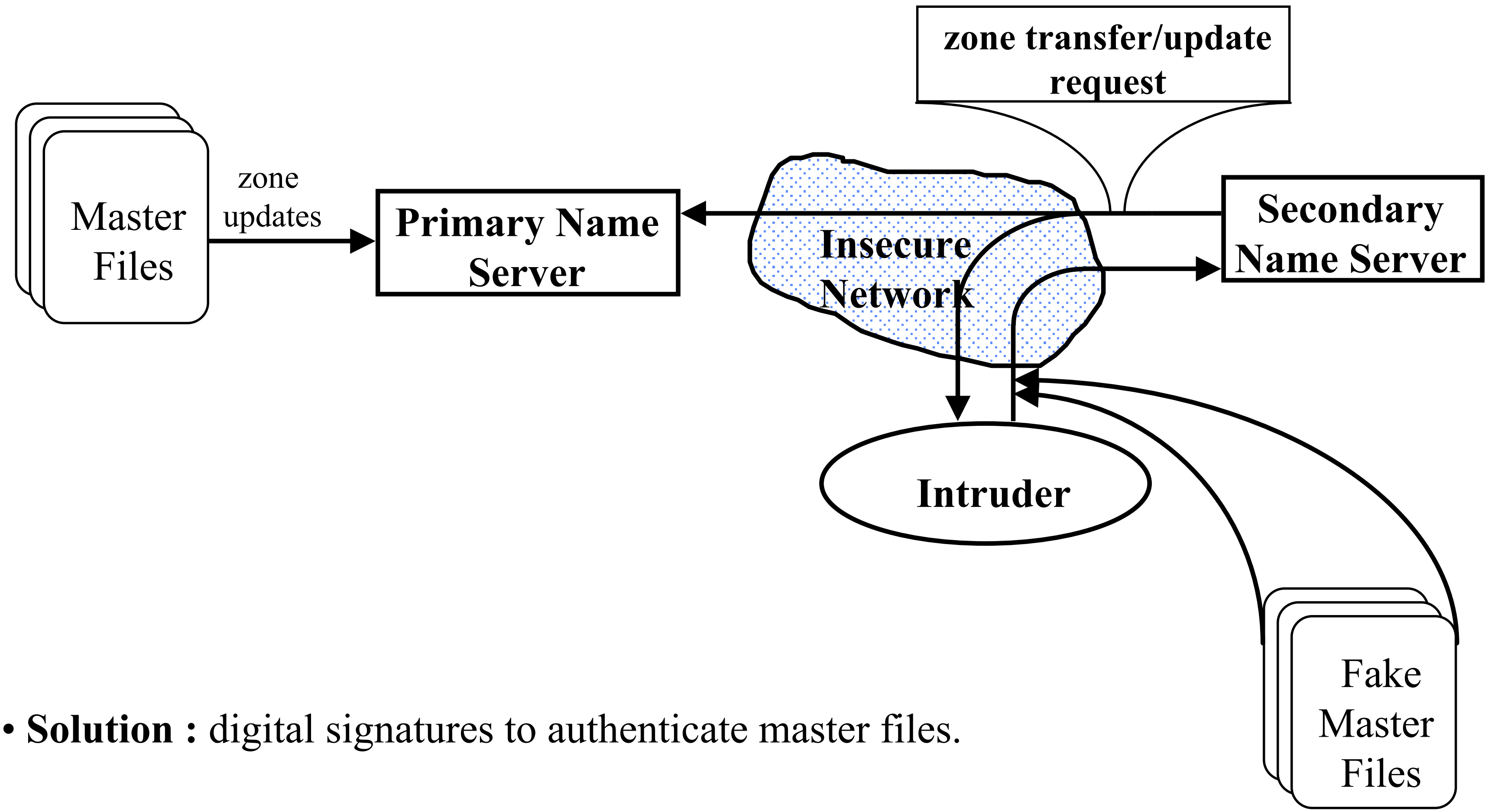
Possible attacks : spoofing



- **Can lead to :**
  - denial of service => intruder claims QNAMEs are inexistent
  - **solution :** NXT RR to authenticate the nonexistence of names or types for existing names.

  - masquerade => intruder indicates his host's address in responses.
  - **solution :** SIG RR to authenticate resource records.

Possible attacks



• **Solution :** digital signatures to authenticate master files.

Possible attacks

- **Scenario :** A NS wants to restrict service (i.e., recursive), only to a specific set of resolvers.
- **Problem :** access control list not provided.

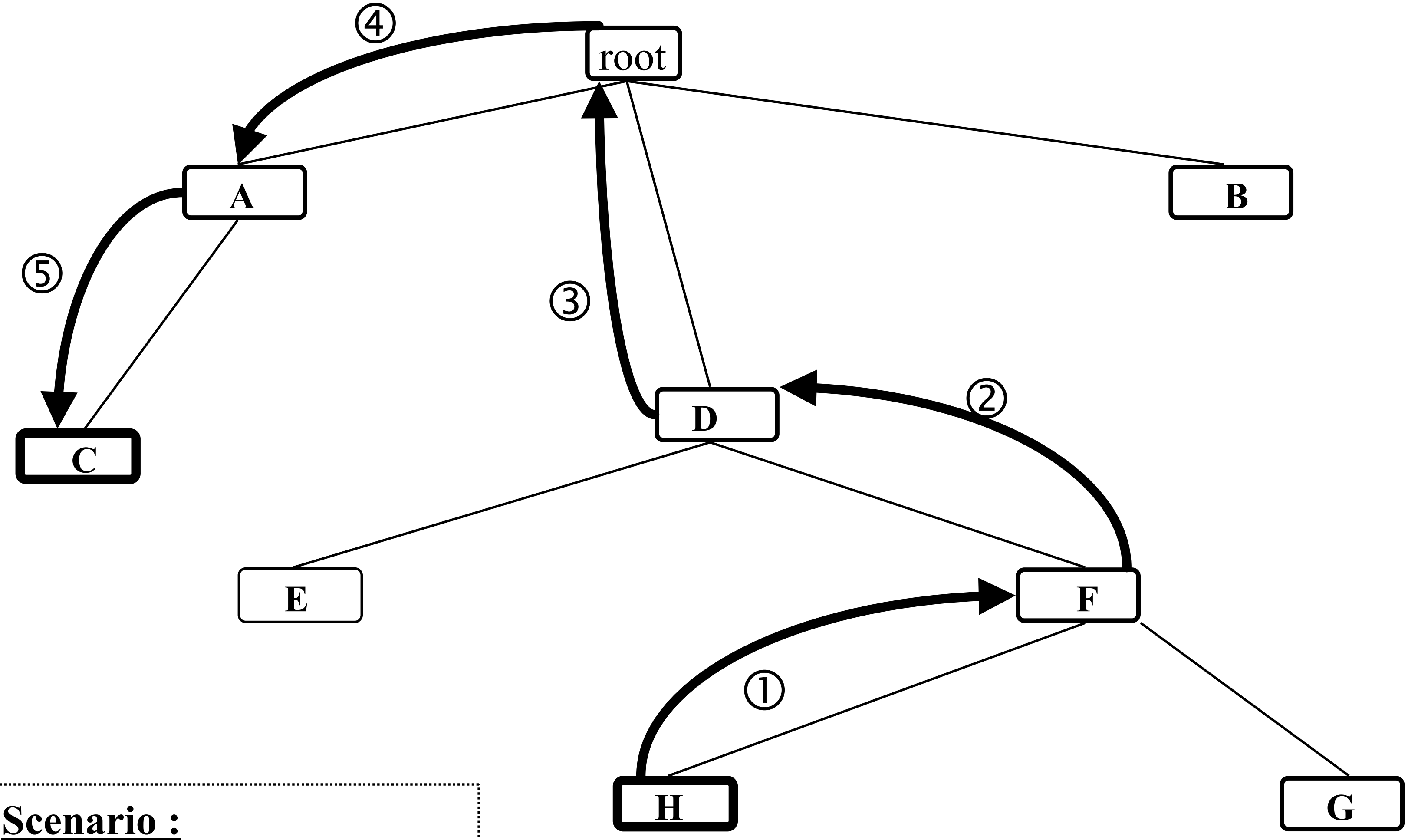
- **Scenario :** An organization wants to maintain the privacy of some names and RRs in its zone.
- **Problem :** anybody can claim to be a secondary NS for that zone and ask for a zone transfer.

- **Solution :** add access control and digital signatures to authenticate transactions and requests (not only RR signatures and reply authentication).

Certificate-like structure in DNS

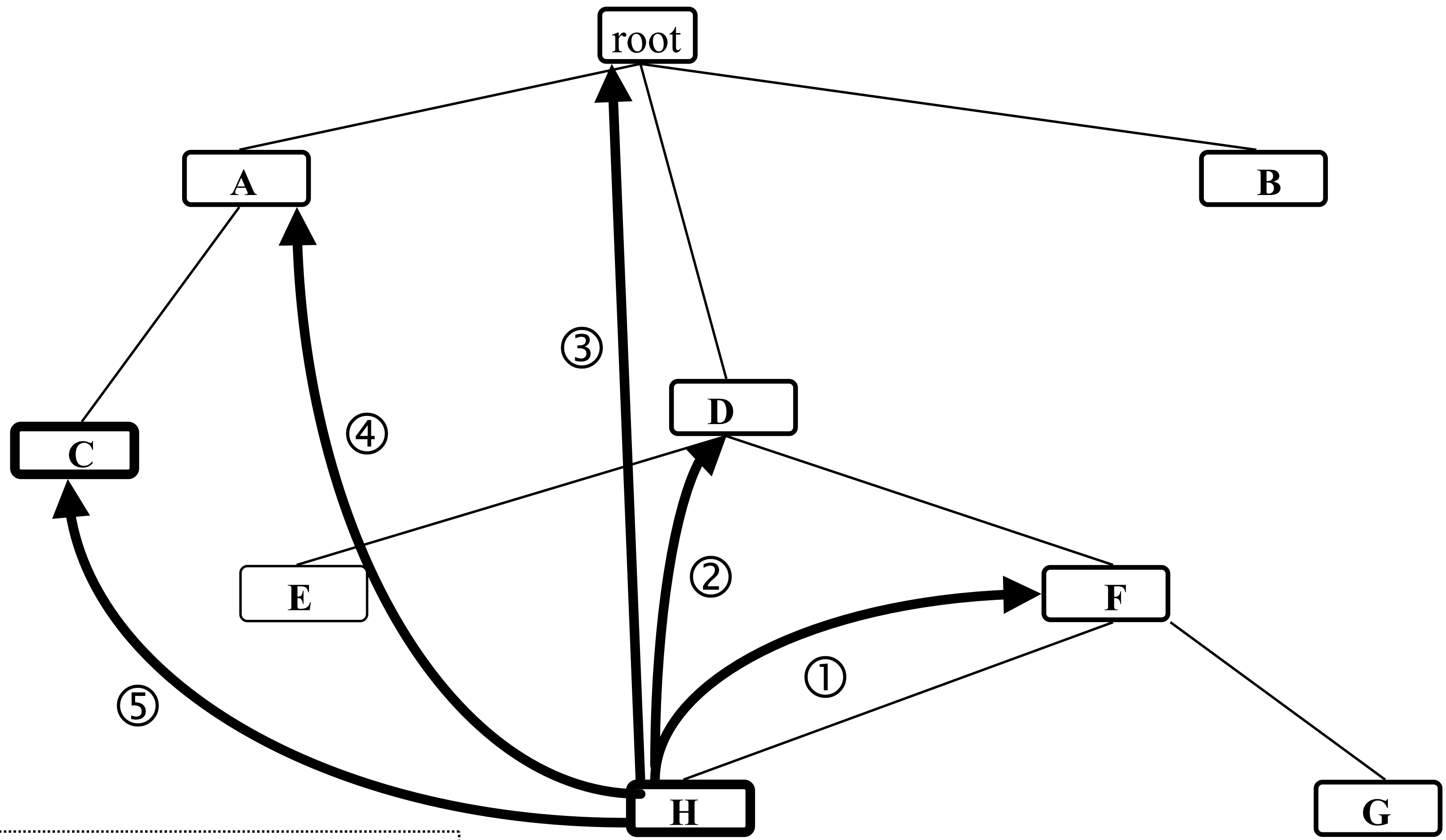
X.509	DNS
version	owner
serial number	labels
algorithm used for signing	algorithm used for signing
issuer	signer's name
validity	signature expiration time
subject	type covered
subject-public-key-info	key footprint
identifiers	time signed
signature	signature

Recursive Trust Hierarchy Traversal in DNS



Scenario :
A query made by a host in the **domain H** for a host in **domain C**.

Iterative Trust Hierarchy Traversal in DNS



Scenario :
A query made by a host in the **domain H** for a host in **domain C**.

Presentation Overview

- **Section 1 :** overview of the extensions, key distribution and data origin authentication.
- **Section 2 :** the KEY (public key) resource record, its structure and use.
- **Section 3 :** the SIG (digital signature) resource record, its structure, use and representation.
- **Section 4 :** the NXT resource record (permits authenticated denial of existence of a name or type in the DNS.
- **Section 5 :** resolver configuration with starting key(s) for secure resolving of DNS requests.
- **Section 6 :** review of operational considerations : key generation, lifetime, and storage.
- **Section 7 :** levels of conformance for resolvers and servers.

Section 1 : Overview of DNS security extensions

- **Services provided :**
  - key distribution
  - data origin authentication
  - transaction and request authentication

- **Services not provided :**
  - access control lists or other means to differentiate inquires
  - confidentiality for queries or responses

Section 1 (continued)

- **Key distribution :**
  - a new KEY RR type defined to hold public keys
  - keys associated with domain names
  - security aware NSs automatically return KEY RRs as additional information, along with the RRs actually requested.
- **Data origin authentication and integrity :**
  - a new SIG RRs type defined to hold digital signatures
  - a single private key that signs for an entire zone
  - the zone private key kept off-line, periodically signs RRs in the zone
  - data origin authentication belongs to a zone not an NS => compromise of a server will not necessarily affect the entire zone
  - resolvers can learn the public keys of zones :
    - by reading it from a DNS
    - by having it statically configured

Section 1 : special considerations

- **TTL :**
  - TTL ticks down when RRs are cached => TTL left out of the signature.
  - an original TTL is included in the signature; it is included in the RR along with current TTL
  - signatures include also a time signed and expiration time

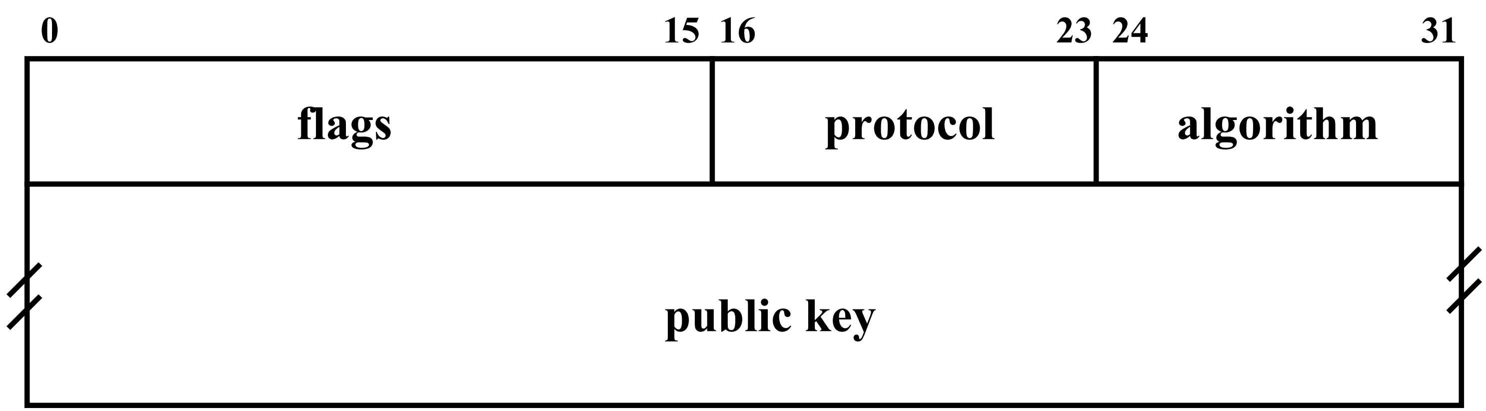
- **Delegation Points :**
  - leaf nodes of a zone (delegation points to a subzone) => viewed as belonging to subzone
  - occur in two master files signed by zone's and subzone's keys
  - KEY RR of the subzone appears in the zone's master file, signed by zone's key
  - NSs and A(glue) RRs for subzone are signed by subzone's key

- **CNAME RRs :**
  - KEY, SIG, and NXT RRs allowed along with CNAME RR
  - suppress CNAME processing for the above types as done on CNAME retrieval
  - automatically return SIG RRs authenticating CNAME RRs

- **DNS Transaction and Request Authentication :**
  - the private key used belongs to the host initiating the transaction/request, not to a zone.

Section 2 : KEY RR

- Used to document a public key associated with a domain name
- Signed by a digital signatures for authentication
- Associated with :
  - a zone
  - a host or other entity
  - an user account
- Format :



Section 2 : KEY RR's Fields

- **Flags :**
  - bits 0, 1 : type field => key used for authentication, confidentiality or not used
  - bit 2 : experimental
  - bits 3, 4 : must be zero
  - bit 5 : indicates that the key is associated with an user or account at an end entity (host)
  - bit 6 : indicates that key is associated with a non-zone entity (usually a host)
  - bit 7 : indicates that key is associated with a zone
  - bit 8 : reserved
  - bit 9 : "e-mail" bit => key used with MIME security multiparts
  - bits 10, 11 : reserved, must be zero
  - bits 11-15 : indicates whether key can sign RRs

- **Protocol :**
  - indicates in conjunction with which protocol the key is used

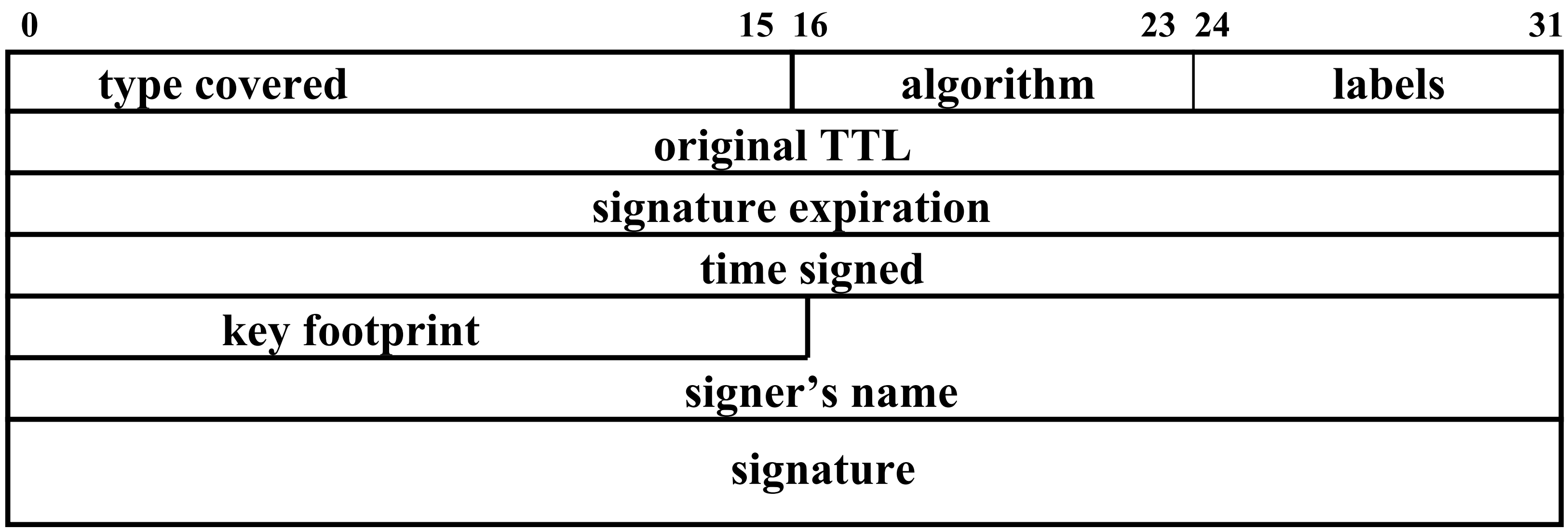
- **Algorithm :**
  - a value of 1 => MD5/RSA algorithm
  - values from 2 through 252 available for assignment to other algorithms

Section 3 : SIG RR

- Authenticates RRs of a particular type, class and name

- Binds the signature to a time interval and the signer's name

- RDATA format :



NOTES :

- labels : count of how many labels there are in the SIG RR owner name excluding "*".
- key footprint :
  - used to select among multiple keys types for same algorithm (e.g., **sig** vs. **auth** keys)
  - its exact meaning is algorithm dependent

Section 3 : canonical form and order for RRs

- **Canonical form and order needed because :**
  - RRs' owner names are stored in upper and lower case
  - RRs' order is not preserved in master files
  - a SIG RR may sign one or more RRs => they need to be ordered and in canonical form

- **Canonical form for RRs :**
  - converted to lower case
  - owner names expanded (not compressed with DNS compression)
  - the original TTL substituted by the current TTL

- **Canonical order for RRs :**
  - labels are ordered as left justified unsigned octets
  - a missing octet sorts before a zero octet
  - names are sorted by starting with the highest level (nearest to the root) label down to the leafs
  - within a particular name, types are sorted similarly to labels
  - SIG RRs signing a type are placed immediately after all the RRs of that type

Section 3 : Other SIG RRs

- **Zone transfer (AXFR) SIG :**
    - used to authenticate zone transfers
    - created by signing an entire zone

- **Transaction and Request SIG :**
    - appended to the end of a response or query, to authenticate the transaction
    - signed by the host’s key not by the zone’s key

NOTE :
Security aware NSs should attempt to send SIG RRs which authenticate the RRs requested, along with those RRs

Section 4 : Non-existent name and type authentication

- The extensions provided so far authenticate only **existing** names/types.

- NXT RR => authenticates the **non-existence** of names or types
  - in a master file all RRs are ordered in canonical order
  - for a name interval in which no name exists a NXT RR is created
    - the owner is the name with which the interval begins
    - the RDATA of NXT RR contains :
      - all existent types for the owner of the NXT RR
      - the name where the name interval ends

- NXT RRs authenticate :
  - the **non-existence of a type** at an existing name => the NXT RR at that name lists all existing types for that name
  - the **non-existence of a name** => the NXT RR for an interval containing that name

- NXT RR that authenticates **a name is the last one** in a zone :
  - name space is considered circular => starts and ends with the zone's name
  - the last NXT RR => the owner is the last name, in the RDATA we have the zone's name

Section 5 : Initial Resolver Configuration

- Resolvers need to be configured with trusted public keys of one or more zones

- Resolver can then learn the public keys of other zones, through glue records

- Greater security is obtained if resolvers configured with keys for all critical zones

- Secure NSs classify data in four classes :
  - authenticated => signatures verified
  - pending => at least one signature the NS tries to verify
  - insecure => data obtained through a non-secure zone
  - bad => signature verification failed

- Two new header bits are used :
  - AD in responses => when set, data was verified by NS that sent it
  - CD in queries => when set, unverified data is acceptable (reduces NS response latency)

- Chaining through zones :
  - security aware NSs should not step from a secure zone to a non-secure one, unless the non-secure zone is certified to be non-secure(through a KEY RR)
  - no zones can be trusted if they can be reached only via non-secure zones.

Section 6 : Operational Considerations

- **Key size** :
  - recommended minimum 640 to 1000 bits

- **Key storage :**
  - zone private keys and zone file master copy to be kept and used off-line.
  - RRs and zones to be authenticated/signed periodically, off-line
  - only one-way information flow from signer machine to the rest of the network

- **Key generation :**
  - recommended to happen of-line

- **Key lifetimes :**
  - zone keys => less than 4 years; recommended 13 month
  - on-line user/entity keys => less than 36 days

- **Signature lifetimes :**
  - small multiple of the TTL

Section 7 : Conformance

1. Server conformance :

- **minimal :**
  - ability to store and retrieve KEY, SIG and NXT resource records
- **full :**
  - ability to read SIG, KEY and NXT RRs in zone files
  - ability to add appropriate SIG and NXT RRs as needed
  - automatic inclusion of SIG, KEY and NXT RRs in responses
  - recognize the CD and use of the AD bit headers as necessary
  - proper handling of NXT RRs at delegation points

2. Resolver conformance :

- **minimal :**
  - ability to handle KEY, SIG and NXT RRs when explicitly requested
- **full :**
  - understand KEY, SIG and NXT resource records
  - maintain proper information in its caches about which RRs have been authenticated
  - perform additional queries as necessary to obtain KEY, SIG and NXT RRs
  - set the CD query bit header in its requests (usually)