Internet Engineering Task Force J. Daley, Ed.
Internet-Draft .nz Registry Services
Intended status: Informational S. Morris
Expires: February 01, 2014 Internet Systems Consortium
J. Dickinson

Sinodun

July 31, 2013

dnsxml - A standard XML representation of DNS data
draft-daley-dnsxml-00

Abstract

This memo describes a syntax for encoding DNS Resource Records in XML, and a schema to define that syntax written in XML Schema. It can be used to represent all DNS RDATA. This can be used by diverse applications as a common format.

DNS Resource Records are represented as XML elements with the name of the element taken from the mnemonic used to represent the DNS Resource Record in presentation format. The RDATA is represented as XML attributes or content of the element. The attribute names are taken from the RDATA field names specified in the normative RFC.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 01, 2014.

Table of Contents

1. Introduction

Historically, DNS Resource Records (RRs) have a presentation format and wire format. The presentation format is typically used to conveniently store DNS RRs in Human Readable Form. The wire format is typically used in transport and communication between DNS protocol elements.

This memo describes an alternative presentation format for DNS using an XML syntax [W3C.REC-xml-20081126] with an XML schema defined in XML Schema [W3C.REC-xmlschema-1-20041028]. These two parts taken together are called dnsxml.

The purpose of dnsxml is to enable XML based applications and protocols to contain DNS Resource Records in their native XML rather than the existing DNS presentation format. This simplifies the processing of XML documents that contain DNS Resource Records and enables the use of standard XML tools such as validation and transformation on those records.

An example of an XML based protocol that may choose to use dnsxml is Extensible Provisioning Protocol (EPP).

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Design goals

dnsxml consists of two parts:

- an XML syntax for representing DNS RRs in XML;
- an XML schema definition of that syntax.

Each of these two parts has a set of design goals, as set out below.

2.1. Design goals for the XML syntax for DNS RRs

The XML syntax should:

1. be clear, unambiguous, succinct and easy to read for the human reader;
2. use as closely as possible the presentation format for RDATA fields given in various RFCs, even if that reduces overall readability for those unfamiliar with DNS, in the expectation that it will be easier to use for those familiar with DNS;
3. split out RDATA into separate components so as to minimise the secondary parsing that an application needs to do in order to obtain the values of individual RDATA fields;
4. be independent of any name server implementation;
5. allow the representation of an RR of unknown type as described in RFC 3597 [RFC3597].

2.2. Design goals for the XML schema definition

The schema definition should:

1. validate as much RDATA as possible;
2. not require excessive processing power for validation;
3. not impose any restrictions on the future definition of a new RR element or a change to an existing RR element;

4. allow for any new RR to be described as an extension of this schema definition and used as easily as any RR element described in it;

5. ensure that a new version of this schema definition may include new RRs or changes to existing RRs that have been described in new RFCs, without preventing the continuing use of any extensions;

6. not require an excessive frequency of updates to address changes in normative RFCs or the IANA registry;

7. support semantic inference between RDATA fields that represent semantically equivalent data.

Clearly, some of these goals need to be balanced against each other.

2.3. Semantic inference

The design goal [semantic] for semantic inference is intended to allow users of dnsxml to carry out semantically-aware processing, which may be achieved through the use of schema-aware XSLT.

2.4. Supported DNS RR tyopes

The following RFCs and Resource Records types are supported in dnsxml:

- From [RFC1035], A, CNAME, HINFO, MB, MG, MINFO, MR, MX, NS, NULL, PTR, SOA, TXT and WKS.
- From [RFC1183], AFSDB, ISDN, RP, RT and X25.
- From [RFC1706], NSAP.
- From [RFC1712], GPOS.
- From [RFC1876], LOC.
- From [RFC2163], PX.
- From [RFC2230], KX.
- From [RFC2538], CERT.
- From [RFC2672], DNAME.
- From [RFC2782], SRV.

- From [RFC2845], TSIG.
- From [RFC2874], A6.
- From [RFC2930], TKEY.
- From [RFC2931], SIG.
- From [RFC3123], APL.
- From [RFC3445], KEY.
- From [RFC3403], NAPTR.
- From [RFC3596], AAAA.
- From [RFC4025], IPSECKEY.
- From [RFC4034], DNSKEY, DS, NSEC and RRSIG.
- From [RFC4255], SSHFP.
- From [RFC4408], SPF.
- From [RFC4431], DLV.
- From [RFC4701], DHCID.
- From [RFC5155], NSEC3 and NSEC3PARAM.

Obsolete DNS resource records are not supported. Neither are the NB and NBSTAT RR types defined in [RFC1002].

2.5. Exclusions and limitations

The focus of dnsxml is DNS data only and dnsxml is not intended as a replacement for the DNS protocol. For this reason there are a number of parts of DNS that are not represented in dnsxml:

- It is not possible to define all the parts of a DNS datagram in dnsxml. There is no XML element in dnsxml that represents the header section of a DNS datagram or the question section.
- There is no representation of the OPT pseudo-RR because OPT, as described in [RFC2671], "pertains to a particular transport level message and not to any actual DNS data"

For clarity:

- No use is made of Master File Format [RFC1035], section 5.1.

- dnsxml is not intended to obsolete the presentation format of RR types as specified in their normative RFCs.

- dnsxml is not intended to limit the presentation formats of future RR types.

3. The XML syntax and XML schema for DNS RRs

These are examples of resource records represented in this syntax:

```
<A owner="example.com." class="IN" ttl="86400" address="192.0.2.1"/>
<TXT>Any text here</TXT>
```

and this is an example of an RRSet:

```
<RRSet owner="example.com." class="IN" type="A" ttl="86400">
  <A address="192.0.2.1"/>
  <A address="192.0.2.2"/>
</RRSet>
```

3.1. General features

3.1.1. Unique XML element for each RR type

Each DNS RR type has a corresponding element. That ensures that the schema definition can constrain the allowable attributes on a per RR basis. It also meets the design goal of clear, unambiguous and easy to read.

3.1.2. Representation of RDATA

Most RDATA is represented in attributes as this significantly reduces the verbosity of the XML. Some RDATA is represented as the content of the element.

3.1.2.1. RDATA represented as XML attributes

For each element that represents an RR type, the attributes specified correspond to those specified in the normative RFC that defines the RDATA for that RR type. For example, the MX element has the specific attributes of 'preference' and 'exchange' as specified in [RFC1035].

Extensive use is made of the XML Schema [W3C.REC-xmlschema-1-20041028] attribute 'use="required"' by which

the use of an attribute in conforming documents is mandated. This is used when the normative RFC for that RR type states that an RDATA field 'MUST' exist.

The type of an attribute is chosen to represent the presentation format for the RDATA field specified in the relevant RFC. For example a field specified as 32 bit unsigned integer is represented using the XML Schema [W3C.REC-xmlschema-2-20041028] type of 'unsignedInt'.

Where there are multiple presentation formats for a single RDATA field, the defined type is a union of two built-in types.

3.1.2.2. RDATA represented as element content

Some RDATA is better suited to be represented as the content of an element rather than as an attribute. The following criteria have been used as a general guide to determine when to use this method fo representation:

- the RDATA is anonymous. In other words the RDATA field is simply labelled as RDATA and no other label is given;
- the RDATA is of variable length or is expected to be long enough that representing it in an attribute will make it hard to read;
- the text representation of the RDATA in the normative RFC allows it to be split across multiple lines.

To aid the implementer the following table lists the elements that allow or require content:

Element	RDATA field	Type	Nillable
APL	-anonymous-	string	yes
NULL	-anonymous-	string	yes
SPF	-anonymous-	string	no
TXT	-anonymous-	string	no
TYPE	-anonymous-	hexWithWhitespace	no
DLV	digest	hexWithWhitespace	no
DS	digest	hexWithWhitespace	no
SSHFP	fingerprint	hexWithWhitespace	no
TKEY	other data	hexWithWhitespace	yes
TSIG	other data	hexWithWhitespace	yes
WKS	bitmap	hexWithWhitespace	no
CERT	certificate or CRL	base64Binary	no
DHCID	-anonymous-	base64Binary	no

DNSKEY	public key	base64Binary	no
IPSECKEY	public key	base64Binary	no
KEY	public key	base64Binary	no
RRSIG	signature	base64Binary	no
SIG	signature	base64Binary	no

Table 1

More information on the types used to represent variable length binary data can be found in Section 3.3.7.

3.1.3. Use of XML Schema

This schema is written using XML Schema [W3C.REC-xmlschema-1-20041028] and [W3C.REC-xmlschema-2-20041028] because this is a W3C standard and provides the necessary level of flexibility to correctly specify the preferred syntax. Other schema languages could have been used just as well.

3.1.4. Use of XML Namespaces

XML Namespaces [W3C.REC-xml-names-20091208] need to be used in the schema to reference the defined types. Any document validated against dnsxml must contain a namespace reference in order for it to validate properly. For example

```
<dnsxml xmlns="urn:ietf:params:xml:ns:dns"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:ietf:params:xml:ns:dns
dnsxml-1.0.xsd">
```

In that example the default namespace is set to refer to elements and attributes from dnsxml. A third party extension could be included in the namespace declarations, with a specified prefix, and so all use of the extension would be clearly identified by use of that prefix. This is described more fully in Section 3.4.

3.2. Elements and RRs

3.2.1. Base RR element and base attributes

All elements that represent RRs are derived from an abstract element.
All elements include the attribute group 'baseAttributes' that provide the 'class', 'owner', 'ttl' and 'rdlength' attributes. The elements that represent RRs are defined using the XML Schema [W3C.REC-xmlschema-1-20041028] feature of substitutionGroup to substitute for the abstract RR element.

This same mechanism is used by any new RR types that are defined in extensions, which ensures they are treated equally to built-in elements rather than needing to appear in a separate extension element. This is covered further in Section 3.4.1

It should be noted that, as this is an abstract element, it cannot be used in an XML document that is to be validated by dnsxml.

3.2.2. RRset element

The schema has an element called 'RRset' that represents an RRset, using the definition from [RFC2136] of a set of RRs that share the same 'owner', 'class' and 'type' RDATA fields, each of which is represented as attributes. In addition a 'ttl' attribute is specified because [RFC2181] requires all the RRs in an RRSet to share the same ttl.

(If an RR type is ever defined with the mnemonic of 'RRSET', this would present future versions of dnsxml with a naming conflict.)

Any element that represents an RR can be used either standalone or within an RRset element.

The RRset element may be empty to represent an empty RRset.

The RRset element implementation in dnsxml has a number of limitations to the validation that it performs. These could theoretically be fixed but would require such significant alterations to the schema that a number of important characteristics, including extensibility, simplicity and ease of use, would be lost.

The validation limitations of the RRset element are:

- An RRset element may contain elements that represent different DNS RR types from the type specified for the RRset element. The processing behaviour of such errant elements is left to the application to decide.

- It is possible for the elements within an RRset element to have 'class', 'owner' and 'ttl' attributes that contradict those of the RRset element. The processing behaviour of such errant elements is left to the application to decide.

- [RFC2136] lists a number of RR types (SOA, WKS and CNAME) that can only appear once in an RRset. This restriction is not enforced in

this schema.

This may mean that the RRset element is used by appplications as a general container for a set of RRs, which is quite different from the normative use of an RRset in DNS.

3.2.3. TYPE element for holding unknown RR types and raw RR data

To fit with the convention of naming the element after the RR type mnemonic it would be preferable to have 65535 different elements with names of the form TYPEnnnnn, but this would make the schema unnecessarily long and slow to process. Instead an element called TYPE is included, named in the spirit of [RFC3597] that can hold an RR of any type. This has an attribute 'rrtype' that holds the DNS type as an unsignedShort (type mnemonics cannot be used here) and the raw data is represented as content of the element. The optional base attribute 'rdlength' can be set if required. See Section 3.3.11 for more information on the 'rdlength' attribute.

No use is made of the special token '\#' specified in [RFC3597] to indicate the start of the RDATA for a TYPE RR as this is superfluous in the XML representation.

To comply with the [RFC3597] specification of the presentation format for an RR of an unknown type, the 'rdata' attribute of the TYPE element is of the type hexBinary.

This element can also be used to contain 'broken' DNS data.

3.2.4. CLASS

The 'class' attribute is a union of three types, allowing three different representation formats:

1. The defined mnemonics of [RFC6195], section 3.2. The mnemonics of NONE, * and ANY are included for completeness;

2. The CLASSnnnnn mnemonic in conformance with [RFC3597], section 5.

3. An integer in the range 0-65535

dnsxml does not set a default of "IN" for CLASS as this would be incorrect for some RR types including TKEY as defined in [RFC2930]. Nor is the 'class' attribute required.

3.2.5. Top level container element

There is an element in the schema called 'dnsxml' that does not represent any DNS data. It is provided as an optional top-level container element, which can be used in a document as the opening element and contain an arbitrary list of 'RRSet' elements and elements representing RRs. However it does not have to be used, as both the 'RRSet' element and the elements representing RRs are declared as top level elements and so can be used directly in a valid document. It would be sensible for the 'dnsxml' element to be used in document that only references this schema (a standalone document), or as a container for a set of elements.

For example, a standalone document might look like this:

```
<dnsxml xmlns="urn:ietf:params:xml:ns:dns"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:ietf:params:xml:ns:dns
dnsxml-1.0.xsd">

  <A owner="example1" class="IN" ttl="86400" address="192.0.2.2" />
  <A owner="example2" class="IN" ttl="86400" address="192.0.2.1" />
</dnsxml>
```

Whereas a fragment of a document where dnsxml is embedded, might look like this:

```
    :
<someApp>
  <someCommand>
    <dns:A owner="example2" class= IN" ttl="86400"
           address="192.0.2.1" />
  </someCommand>
</someApp>
    :
```

3.3. Attributes and RDATA

3.3.1. Semantic equivalence of RDATA

Fields that share the same semantic use (for example an IP address or domain name) use the same defined types or the types that are derived from a common type, in order to enable later semantic inferences to be developed.

3.3.2. Anonymous RDATA

The SPF, TXT and DHCID RR types have a single anonymous RDATA field just referred to as the RDATA in the normative RFC. For each of these the attribute that represents the RDATA is called 'rdata'.

3.3.3. IP addresses in RDATA

All attributes that contain IPv4 address are defined to be of type 'ip4AddressType', which uses a regular expression to validate that the content of the attribute is a valid IPv4 address. Attributes that hold IPv6 addresses are similarly defined to be of type 'ip6Addresstype', which also uses a regular expression to validate the content of the attribute.

In addition the type 'ipAddressType' exists as a union of 'ip4AddressType' and 'ip6AddressType' for use in the APL RR type.

3.3.4. Domain names in RDATA

Attributes for RDATA fields that are used for domain names are all of the type 'domainType'. This is defined to be a 'string' with the maximum length restricted. A later development for a future version may be to validate the contents of these attributes using a regular expression.

3.3.5. XML in RDATA

Any data in attributes that represent an RDATA field that can contain XML MUST be escaped using the rules given in [W3C.REC-xml-20081126]

Because escaping is a standard part of XML, no specific type is defined to use for those fields where escaping may be required.

3.3.6. Unparsed data in RDATA

A number of RDATA fields are defined in RFCs as containing any text data. Any data in the attributes that represent these RDATA fields MUST be escaped following the rules given in [W3C.REC-xml-20081126]

3.3.7. Variable length binary data in RDATA

There are a number of examples where RDATA contains a binary field such as set of flags or a bit map field. For example WKS has a variable length bit map field, with no defined presentation format. These fields are represented either by the defined type of 'hexWithWhitespace'or the built-in type of 'base64Binary' depending on context. XML Schema [W3C.REC-xmlschema-2-20041028] in turn references [RFC2045] for the definition of base64. The built-in type of 'hexBinary' is not suitable because it does not allow whitespace,

whereas the presentation format of many RR types does for hexadecimally presented RDATA.

It should be noted that the NSEC3 RR type has the Next Hashed Owner Name field that may be up to 255 octets long but whitespace in the presentation format is forbidden and so a value that is 255 octets long will have readability problems whether it is an attribute or element content. For simplicity this is encoded as an attribute of defined type 'base32HexRestricted' that uses a regular expression to validate the allowable characters.

3.3.8. Preferences in RDATA

A number of RR types have a preference RDATA field, namely KX, MX, PX, RT, NAPTR. The attributes that represent the preference field for these RR types are all defined to be of the type 'preferenceType' on the potentially contentious grounds that they are semantically equivalent.

Additionally the IPSECKEY RR type has a precedence RDATA field, which is defined as being semantically equivalent to the preference RDATA field of the MX RR type. The attribute representing this field is therefore also defined as being of type 'preferenceType'.

3.3.9. Seconds (units of time) in RDATA

Many RDATA fields are defined as unsigned integers that record a number of seconds. There are a number of different types of time field:

- Fields such as the 'refresh' field of the SOA RR type, are defined as an interval. The attributes that represent these fields are defined as being of type 'secondsInterval32Type'.

- Fields such as the 'signature expiration' field of the RRSIG RR type, contain the number of seconds since the unix Epoch. This is in turn comes in a number of variants:

  * The 'timesigned' field of the TSIG RR type, which has the wire format of a 48 bit unsigned integer and the corresponding attribute is defined as being of type 'secondsSinceEpoch48Type';.

  * Those that use a 32 bit unsigned integer and so are defined as being of type 'secondsSinceEpoch32Type', which is a restriction of 'secondSinceEpoch48Type'

* Those that use a 32 bit unsigned integer but whose presentation format also allows a text representation of the form 'YYYYMMDDHHmmSS' such as the 'signatureexpiration' field of RRSIG. These are defined as being of type 'secondsSinceEpochTextType', which is a union of 'secondsSinceEpoch32Type' and a 14 character string.

o Fields such as the 'ttl' field of all RR types, contain an interval but with specific semantic usage of Time To Live.

Semantic equivalence is maintained by all the time types being derived from a common type 'secondsBaseType'.

3.3.10. RCODE in RDATA

Attributes that represent an RCODE are either of type 'rcode16Type' or 'rcode12Type' depending on the number of bits in the corresponding RDATA field. These are all derived from the 'baseRcode16Type' to provide semantic equivalence.

3.3.11. RDATA field that specifies the length of another RDATA field

The Resource Record format as defined in [RFC1035] includes an 'rdlength' field. There is a corresponding base attribute called 'rdlength' that is optional. This attribute is of type 'rdataLengthType', which is limited to an unsigned 16 bit integer.

Numerous RR types including NSEC3, TKEY and TSIG have an RDATA field that specifies the length of another RDATA field in octets. The attributes that represent these fields all share the same type of 'rdataLengthType', or 'rdataLength8Type', the latter being limited to an unsigned 8 bit integer.

It could be argued that RDATA fields that hold the length of other RDATA fields do not need to be included in dnsxml as these values can be calculated directly from the data with certainty. However these fields have been included for completeness and for unknown future uses, but they are generally defined as 'use="optional"' to allow for applications that will calculate the length directly.

3.3.12. Mnemonics for integer RDATA

A number of RR types, for example DNSKEY, RRSIG, DS and DLV, have fields in their RDATA that are integer types but also have string mnemonics. The attributes that represent these fields are defined as a union of two simple types, one that allows integer representation and one that allows a string representation. The string representation is restricted to the known mnemonics but the integer values are not restricted to those for which a mnemonic is defined.

A number of sets of mnemonics are defined in the IANA registry [dns-sec-alg-numbers]. If a new mnemonic is defined by IANA after the definition of this protocol, a new version of dnsxml will need to be issued for that to be incorporated into the schema. Until that time the mnemonic will fail validation and instead the integer the mnemonic refers must be used or the TYPE syntax of [RFC3597].

Mnemonics are only defined in the schema where they appear in a normative RFC and not where they appear in an online database, such as the allowable values of 'host' and 'cpu' in the HINFO RR type.

Various RFCs previously referenced have been used as the normative references for the lists of mnemonics and in addition to those [RFC4509], [RFC5702], [RFC5933] and [RFC6605], have been used for DNSSEC algorithm mnemonics. [RFC6195] has been used as the normative reference for the mnemonics for 'class' and 'rcode'.

3.3.13. Cryptographic algorithms and digest types in RDATA

There are two sets of cryptographic algorithms and digest types specified in RDATA:

- Those specified for DNSSEC RFCs. The CERT RR type algorithm type references the DNSSEC types for its RDATA. The attributes that represent this RDATA are defined to be of defined type 'dnssecAlgorithmType'.

- Those specified in [RFC4255]for SSHFP. The terminology is also different with the 'fptype' RDATA of SSSHFP being semantically equivalent to the 'digest type' RDATA of DNSSEC RR types. The attributes that represent this RDATA are defined to be of types 'sshAlgorithmType' and 'sshDigestType'.

These attributes that represent these different RDATA are derived from the common base type of 'baseAlgorithmType', preserving semantic equivalence.

3.3.14. RDATA of the KEY and SIG RRs

The KEY and SIG RRs are unusual in that their wire formats are identical to other RR types (DNSKEY and RRSIG respectively) but their allowable values are different. This leads to some notable design decisions for dnsxml:

- The 'flags' RDATA fields of KEY and DNSKEY are both functionally equivalent as they flag the use of the key material, but the allowable values are different as [RFC4034] allows bit 15 to be set in DNSKEY, whereas [RFC3445] forbids that for KEY. Given this divergence, the two 'flags' attributes are both defined as being of type 'unsignedShort' rather than sharing a common defined type to allow for semantic inference.

- While the 'protocol' RDATA field for both the KEY and DNSKEY RR types are currently semantically and functionally identical the corresponding attributes do not use a common defined type for either semantic or functional equivalence. This decision is taken because the two fields are defined independently and so may diverge as the 'flags' fields have done.

- The 'type covered' RDATA field of SIG was originally used to hold an RR type. The combination of [RFC2931] and [RFC4034] changes this field to only have the allowable value of 0. It is the understanding of the authors that this changes means that this field no longer represents an RR type. Consequently the attribute 'typecovered' in the SIG element is defined as being of type 'unsignedShort' and no semantic link is made with any other attribute that holds an RR type, nor can an RR type mnemonic be used as a value for this attribute.

3.3.15. Lists of RR types in RDATA

A number of RR types including RRSIG and NSEC have RDATA that contains a list of RR types. This is implemented as a list of RR type mnemonics using the XML Schema 'list' feature. The TYPE representation as specified in [RFC3597], section 5 is fully supported.

3.3.16. RDATA of the APL RR type

The APL RR type [RFC3123] is unusual as the representation format specified is a complex encoding of the RDATA whereby the RDATA fields appear in a different order from the wire format and additional separator characters are used. To address this complexity, the APL element provides for two different mechanisms to specify RDATA:

1. using individual attributes that correspond to the individual

RDATA fields; or

2. using a single 'rdata' attribute that contains the textual representation specified in [RFC3123]

To enable these two different mechanisms, the various attributes are optional and so it may be possible for attributes to be ommitted or for the two different mechanisms to be used simultaneously. It is left to the application to decide what action to take in either of these cases.

It should be noted that the 'afdpart' attribute does not fully correspond to the wire format of the RDATA field that it represents. The wire format specification is for only the octets covered by the 'prefixlength' to be present, whereas the attibute requires a full and valid IPv4 or IPv6 address.

It should be noted that the 'n' attribute, if it appears, can only contain the '!' character.

3.3.17. Imprecise RFCs on signed/unsigned integers in RDATA

Some RFCs are not clear on whether a specified RDATA field is a signed or unsigned integer. This syntax has made a reasoned choice. For example the 'refresh' field within the SOA RR type definition in [RFC1035]is not explicitly defined as signed or unsigned, but it would not make sense if a signed integer was used here.

3.3.18. Dependency rules in RDATA

There is no validation of the dependency rules that specify that the value set in one RDATA field limits or specifies the allowable values that may appear in another field of the same RR. For example, as defined for the IPSECKEY RR type.

3.4. Extending the schema

3.4.1. The extension mechanism

All elements that represent RRs are specified using the same mechanism and this is available for the development of third-party extensions.

The schema defines an abstract element called 'RR'. Being abstract, the element 'RR' cannot be instantiated; it is just a placeholder that is designed to be replaced by elements that represent DNS RR. The definition of RR is as follows

```
<element name="RR" abstract="true" />
```

To create an element that represents a new RR type the type for that element is first be created. This is done in one of two ways depending on whether or not the RDATA is to be represented solely in attributes.

If the RDATA is to be represented solely in attributes then the type for the element is defined as a 'complexType' that contains the relevant attributes. The following example is the type for the A element:

```
<complexType name="AType">
  <attributeGroup ref="dns:baseAttributes"/>
  <attribute name="address" type="dns:ip4AddressType"
             use="required"/>
</complexType>
```

If one field of the RDATA is to be represented as content of the element then the type for the attribute is defined as a 'complexType' that contain 'simpleContent' that determines the type of the content and the list of attributes. The following example is the type for the TXT element:

```
<complexType name="TXTType">
  <simpleContent>
    <extension base="string">
      <attributeGroup ref="dns:baseAttributes"/>
    </extension>
  </simpleContent>
</complexType>
```

All the 'simpleContent' in dnsxml is an extenstion of 'string', 'base64Binary' or 'hexWithWhitespace' as listed in Table 1.

The examples above show that the base attributes of 'class', 'ttl', 'owner' and 'rdlength' are included in the element type definition by the inclusion of the attribute group named 'baseAttributes'.

All elements that represent RRs are then defined using the substitutionGroup syntax of XML Schema [W3C.REC-xmlschema-1-20041028] and referencing the newly defined type.

For example, the A element is defined in exactly this manner:

```
<element name="A" substitutionGroup="dns:RR" type="dns:AType"/>
```

This memo defines a number of rules for creating extension:

1. The element representing the new RR type MUST include the attribute group 'baseAttributes'. This is true even if 'class' and 'ttl' attributes are meaningless as they for SIG(0).

2. All RDATA fields MUST be represented.

3. The attributes that represent the RDATA of the new RR MUST reuse existing types wherever possible and where new types are created, every effort SHOULD be made to maintain semantic equivalence.

3.4.2. Creating an extension

The purpose of an extension is to provide syntax for a DNS RR type that is not included in dnsxml. Extensions are specified in a new XML Schema instance document, which has the following characteristics:

- declares its own XML Namespace [W3C.REC-xml-names-20091208];
- references dnsxml both as a namespace and importing that schema;
- uses the extension mechanism to create a new element to represent an RR as described in Section 3.4.1.

An extension schema to add an element representing a new RR called EXAMPLE where all the RDATA is represented in attributes, would look as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.example.org/example-schema"
        xmlns="http://www.w3.org/2001/XMLSchema"
        xmlns:ex="http://www.example.org/example-schema"
        xmlns:dns="urn:ietf:params:xml:ns:dns">

  <annotation>
    <documentation>Example extension to dnsxml</documentation>
  </annotation>

  <import namespace="urn:ietf:params:xml:ns:dns"
          schemaLocation="dnsxml-1.0.xsd"/>

  <element name="EXAMPLE" substitutionGroup="dns:RR"
           type="ex:EXAMPLEType" />

  <complexType name="EXAMPLEType">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="exampleRDATA" type="dns:domainType" />
  </complexType>

</schema>
```

If the RR type is

3.4.3. Using an extension

With an extension declared as described in Section 3.4.2 it can then be referenced in a XML document that also references dnsxml. The use of namespaces will keep the references separate.

```
<dnsxml xmlns="urn:ietf:params:xml:ns:dns"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:ietf:params:xml:ns:dns dnsxml-1.0.xsd"
      xmlns:ex="http://www.example.org/example-schema"
      xsi:schemaLocation=
           "http://www.example.org/example-schema example.xsd">

  <ex:EXAMPLE owner="example" class="IN" ttl="86400"
              exampleRDATA="whatever.tld" />

  <A owner="example" class="IN" ttl="86400" address="192.0.2.1" />

</dnsxml>
```

3.5. Implementing new versions of the schema

If a new version of the schema is developed that includes within it new RR types already described in third party extensions, the use of XML Namespaces [W3C.REC-xml-names-20091208] will ensure that the third party extension can continue to be used.

If a new version of dnsxml were now available and an XML document updated to use that, then the document would still validate correctly. If the author then wanted to use the 'example' RR from the new version of dnsxml as well as the version from the extension then they could do so as it sits in a different namespace.

3.5.1. Use of version specific namespaces

This memo specifies two URNs that can be used to refer to dnsxml. The first of these is a version independent reference 'urn:ietf:params:xml:ns:dns', the second is a version specific reference 'urn:ietf:params:xml:ns:dns-1.0'. A document can use either reference, depending on need.

4. Full schema definition

In the following schema definition a number of regular expressions have been split across multiple lines to enable them to be included in this memo. To use this schema correctly these regular expressions must be combined back into a single line without whitespace or they will not work correctly.

```
<?xml version="1.0"?>
<schema targetNamespace="urn:ietf:params:xml:ns:dns"
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:dns="urn:ietf:params:xml:ns:dns">

  <annotation>
    <documentation>dnsxml v1.0</documentation>
  </annotation>

  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- dnsxml element                        -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="dnsxml">
    <complexType>
      <choice maxOccurs="unbounded">
        <element ref="dns:RRset"/>
        <element ref="dns:RR"/>
      </choice>
```

```
    </complexType>
  </element>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- Base attribute set                    -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <attributeGroup name="baseAttributes">
    <attribute name="owner" type="dns:domainType"/>
    <attribute name="class" type="dns:classType"/>
    <attribute name="ttl" type="dns:ttlType"/>
    <attribute name="rdlength" type="dns:rdataLengthType"/>
  </attributeGroup>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- RRset element                         -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="RRset" type="dns:RRsetType"/>

  <complexType name="RRsetType">
    <choice minOccurs="0" maxOccurs="unbounded">
      <element ref="dns:RR"/>
    </choice>
    <attribute name="class" type="dns:classType"/>
    <attribute name="type" type="dns:rrTypeType" use="required"/>
    <attribute name="owner" type="dns:domainType"/>
    <attribute name="ttl" type="dns:ttlType"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- RR abstract element                   -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="RR" abstract="true"/>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- String encoding types                 -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <simpleType name="hexWithWhitespace">
    <restriction base="string">
      <pattern value="([0-9a-fA-F\s])*"/>
    </restriction>
```

```
   </simpleType>

   <simpleType name="base32HexRestricted">
     <restriction base="string">
       <pattern value="([0-9a-vA-V])*"/>
     </restriction>
   </simpleType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- Basic attribute types                   -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <!-- All the known RRs plus the TYPE
        representation of [RFC3597] (5) -->
   <simpleType name="rrMnemonicType">
     <union>
       <simpleType>
         <restriction base="string">
           <enumeration value="A"/>
           <enumeration value="A6"/>
           <enumeration value="AAAA"/>
           <enumeration value="AFSDB"/>
           <enumeration value="APL"/>
           <enumeration value="CERT"/>
           <enumeration value="CNAME"/>
           <enumeration value="DHCID"/>
           <enumeration value="DLV"/>
           <enumeration value="DNAME"/>
           <enumeration value="DNSKEY"/>
           <enumeration value="DS"/>
           <enumeration value="GPOS"/>
           <enumeration value="HINFO"/>
           <enumeration value="IPSECKEY"/>
           <enumeration value="ISDN"/>
           <enumeration value="KEY"/>
           <enumeration value="KX"/>
           <enumeration value="LOC"/>
           <enumeration value="MB"/>
           <enumeration value="MG"/>
           <enumeration value="MGINFO"/>
           <enumeration value="MR"/>
           <enumeration value="MX"/>
           <enumeration value="NAPTR"/>
           <enumeration value="NS"/>
           <enumeration value="NSAP"/>
           <enumeration value="NSEC"/>
           <enumeration value="NSEC3"/>
```

```
          <enumeration value="NSEC3PARAM"/>
          <enumeration value="NULL"/>
          <enumeration value="OPT"/>
          <enumeration value="PTR"/>
          <enumeration value="PX"/>
          <enumeration value="RP"/>
          <enumeration value="RRSIG"/>
          <enumeration value="RT"/>
          <enumeration value="SSHFP"/>
          <enumeration value="SIG"/>
          <enumeration value="SOA"/>
          <enumeration value="SPF"/>
          <enumeration value="SRV"/>
          <enumeration value="TKEY"/>
          <enumeration value="TSIG"/>
          <enumeration value="TXT"/>
          <enumeration value="WKS"/>
          <enumeration value="X25"/>
        </restriction>
      </simpleType>
      <simpleType>
        <restriction base="string">
          <!-- The following regex need to be reassmebled
               on one line, without whitespace, before use -->
          <pattern
            value="TYPE(6553[0-5]|655[0-2][0-9]\d|65[0-4](\d){2}|
6[0-4](\d){3}|[1-5](\d){4}|[1-9](\d){0,3})"
          />
        </restriction>
      </simpleType>
    </union>
  </simpleType>

  <simpleType name="rrTypeType">
    <union memberTypes="dns:rrMnemonicType unsignedShort"> </union>
  </simpleType>

  <!-- A list of any number of the full RR types -->
  <simpleType name="rrListType">
    <list itemType="dns:rrMnemonicType"/>
  </simpleType>

  <simpleType name="domainType">
    <restriction base="string">
      <maxLength value="255"/>
    </restriction>
  </simpleType>
```

```
  <simpleType name="classType">
    <union>
      <simpleType>
        <restriction base="string">
          <enumeration value="IN"/>
          <enumeration value="CH"/>
          <enumeration value="HS"/>
          <enumeration value="NONE"/>
          <enumeration value="*"/>
          <enumeration value="ANY"/>
        </restriction>
      </simpleType>
      <simpleType>
        <restriction base="string">
          <!-- The following regex need to be reassmebled
               on one line, without whitespace, before use -->
          <pattern
            value="CLASS(6553[0-5]|655[0-2][0-9]\d|65[0-4](\d){2}|
6[0-4](\d){3}|[1-5](\d){4}|[1-9](\d){0,3})"
          />
        </restriction>
      </simpleType>
      <simpleType>
        <restriction base="unsignedShort"/>
      </simpleType>
    </union>
  </simpleType>

  <simpleType name="rcodeMnemonicsType">
    <restriction base="string">
      <enumeration value="NoError"/>
      <enumeration value="FormErr"/>
      <enumeration value="ServFail"/>
      <enumeration value="NXDomain"/>
      <enumeration value="NotImp"/>
      <enumeration value="Refused"/>
      <enumeration value="YXDomain"/>
      <enumeration value="YXRRSet"/>
      <enumeration value="NXRRSet"/>
      <enumeration value="NotAuth"/>
      <enumeration value="NotZone"/>
      <enumeration value="BADVERS"/>
      <enumeration value="BADSIG"/>
      <enumeration value="BADKEY"/>
      <enumeration value="BADTIME"/>
      <enumeration value="BADMODE"/>
      <enumeration value="BADNAME"/>
      <enumeration value="BADALG"/>
```

```
      <enumeration value="BADTRUC"/>
    </restriction>
  </simpleType>

  <simpleType name="baseRcode16Type">
    <restriction base="unsignedShort"/>
  </simpleType>

  <simpleType name="baseRcode12Type">
    <restriction base="dns:baseRcode16Type">
      <maxInclusive value="4096"/>
    </restriction>
  </simpleType>

  <simpleType name="rcode16Type">
    <union memberTypes="dns:baseRcode16Type dns:rcodeMnemonicsType"/>
  </simpleType>

  <simpleType name="rcode12Type">
    <union memberTypes="dns:baseRcode12Type dns:rcodeMnemonicsType"/>
  </simpleType>

  <!-- Only used once but sufficiently generic to get its own type -->
  <simpleType name="serialType">
    <restriction base="unsignedInt"/>
  </simpleType>

  <!-- Only used once but sufficiently generic to get its own type -->
  <simpleType name="idType">
    <restriction base="unsignedShort"/>
  </simpleType>

  <simpleType name="preferenceType">
    <restriction base="unsignedShort"/>
  </simpleType>

  <simpleType name="rdataLengthType">
    <restriction base="unsignedShort"/>
  </simpleType>

  <simpleType name="rdataLength8Type">
    <restriction base="dns:rdataLengthType">
      <maxInclusive value="255"/>
    </restriction>
  </simpleType>

  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- Time related attribute types      -->
```

```
   <!-- - - - - - - - - - - - - - - - - - - - -->

   <!-- A base seconds type that should not be
        used except to derive other types -->
   <simpleType name="baseSecondsType">
     <restriction base="unsignedLong"/>
   </simpleType>

   <simpleType name="secondsInterval32Type">
     <restriction base="dns:baseSecondsType">
       <maxInclusive value="4294967295"/>
       <!-- 2^32-1 -->
     </restriction>
   </simpleType>

   <simpleType name="secondsSinceEpoch48Type">
     <restriction base="dns:baseSecondsType">
       <maxInclusive value="281474976710655"/>
       <!-- 2^48-1 -->
     </restriction>
   </simpleType>

   <simpleType name="secondsSinceEpoch32Type">
     <restriction base="dns:secondsSinceEpoch48Type">
       <maxInclusive value="4294967295"/>
       <!-- 2^32-1 -->
     </restriction>
   </simpleType>

   <simpleType name="secondsSinceEpochTextType">
     <union>
       <simpleType>
         <restriction base="string">
           <maxLength value="14"/>
           <minLength value="14"/>
         </restriction>
       </simpleType>
       <simpleType>
         <restriction base="dns:secondsSinceEpoch32Type"/>
       </simpleType>
     </union>
   </simpleType>

   <simpleType name="ttlType">
     <restriction base="dns:secondsInterval32Type"/>
   </simpleType>

   <!-- - - - - - - - - - - - - - - - - - - - -->
```

```
   <!-- Address related attribute types   -->
   <!-- - - - - - - - - - - - - - - - - - -->

   <simpleType name="ip4AddressType">
     <restriction base="string">
       <!-- The following regex need to be reassmebled
            on one line, without whitespace, before use -->
       <pattern
         value="((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(25[0-5]|
 2[0-4][0-9]|[01]?[0-9][0-9]?)"
       />
     </restriction>
   </simpleType>

   <simpleType name="ip6AddressType">
     <restriction base="string">
       <!-- The following regex need to be reassmebled
            on one line, without whitespace, before use -->
       <pattern
         value="((([0-9A-Fa-f]{1,4}:){7}([0-9A-Fa-f]{1,4}|:))|
 (([0-9A-Fa-f]{1,4}:){6}(:[0-9A-Fa-f]{1,4}|((25[0-5]|2[0-4]\d|1\d\d|
 [1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]
 {1,4}:){5}(((:[0-9A-Fa-f]{1,4}){1,2})|:((25[0-5]|2[0-4]\d|1\d\d|
 [1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]
 {1,4}:){4}(((:[0-9A-Fa-f]{1,4}){1,3})|((:[0-9A-Fa-f]{1,4})?:((25[0-5]|
 2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
{3}))|:))|
 (([0-9A-Fa-f]{1,4}:){3}(((:[0-9A-Fa-f]{1,4}){1,4})|((:[0-9A-Fa-f]
{1,4})
 {0,2}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|
 [1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){2}(((:[0-9A-Fa-f]{1,4})
{1,5})|
 ((:[0-9A-Fa-f]{1,4}){0,3}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)
 (\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){1}
 (((:[0-9A-Fa-f]{1,4}){1,6})|((:[0-9A-Fa-f]{1,4}){0,4}:((25[0-5]|
 2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
 {3}))|:))|(:(((:[0-9A-Fa-f]{1,4}){1,7})|((:[0-9A-Fa-f]{1,4}){0,5}:
 ((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|
 [1-9]?\d)){3}))|:)))(%.+)?\s*"
       />
     </restriction>
   </simpleType>

   <simpleType name="ipAddressType">
     <union memberTypes="dns:ip4AddressType dns:ip6AddressType"/>
   </simpleType>

   <simpleType name="prefixLengthType">
     <restriction base="unsignedByte">
       <maxInclusive value="128"/>
     </restriction>
```

```
   </simpleType>

   <!-- Only used once but sufficiently generic to get its own type -->
   <simpleType name="portType">
     <restriction base="unsignedShort"/>
   </simpleType>

   <!-- Only used once but sufficiently generic to get its own type -->
   <simpleType name="ipProtocolType">
     <restriction base="unsignedByte"/>
   </simpleType>

   <!-- - - - - - - - - - - - - - - - - - - - -->
   <!-- Crypto related attribute types     -->
   <!-- - - - - - - - - - - - - - - - - - - - -->

   <!-- A base algorithm type that should not
        be used except to derive other types -->
   <simpleType name="baseAlgorithmType">
     <restriction base="unsignedByte"/>
   </simpleType>

   <simpleType name="dnssecAlgorithmType">
     <union>
       <simpleType>
         <restriction base="string">
           <enumeration value="RSAMD5"/>
           <enumeration value="DH"/>
           <enumeration value="DSA"/>
           <enumeration value="RSASHA1"/>
           <enumeration value="DSA-NSEC3-SHA1"/>
           <enumeration value="RSASHA1-NSEC3-SHA1"/>
           <enumeration value="RSASHA256"/>
           <enumeration value="RSASHA512"/>
           <enumeration value="ECC-GOST"/>
           <enumeration value="ECDSAP256SHA256"/>
           <enumeration value="ECDSAP384SHA384"/>
           <enumeration value="INDIRECT"/>
           <enumeration value="PRIVATEDNS"/>
           <enumeration value="PRIVATEOID"/>
         </restriction>
       </simpleType>
       <simpleType>
         <restriction base="dns:baseAlgorithmType"/>
       </simpleType>
     </union>
   </simpleType>
```

```
   <simpleType name="sshAlgorithmType">
     <union>
       <simpleType>
         <restriction base="string">
           <enumeration value="RSA"/>
           <enumeration value="DSS"/>
         </restriction>
       </simpleType>
       <simpleType>
         <restriction base="dns:baseAlgorithmType"/>
       </simpleType>
     </union>
   </simpleType>

   <simpleType name="sshDigestType">
     <union>
       <simpleType>
         <restriction base="string">
           <enumeration value="SHA-1"/>
         </restriction>
       </simpleType>
       <simpleType>
         <restriction base="dns:baseAlgorithmType"/>
       </simpleType>
     </union>
   </simpleType>

   <simpleType name="certificateTypeType">
     <union>
       <simpleType>
         <restriction base="string">
           <!-- [RFC2538] (2.1) -->
           <enumeration value="PKIX"/>
           <enumeration value="SPKI"/>
           <enumeration value="PGP"/>
           <enumeration value="URI"/>
           <enumeration value="OID"/>
         </restriction>
       </simpleType>
       <simpleType>
         <restriction base="unsignedShort"/>
       </simpleType>
     </union>
   </simpleType>

   <simpleType name="keytagType">
     <restriction base="unsignedShort"/>
   </simpleType>
```

```
   <!-- - - - - - - - - - - - - - - - - - - - -->
   <!-- Miscellaneous attribute types     -->
   <!-- - - - - - - - - - - - - - - - - - - - -->

   <simpleType name="longitudeType">
     <restriction base="string"/>
   </simpleType>

   <simpleType name="latitudeType">
     <restriction base="string"/>
   </simpleType>

   <simpleType name="altitudeType">
     <restriction base="string"/>
   </simpleType>

   <simpleType name="gatewayType">
     <union>
       <simpleType>
         <restriction base="dns:domainType"/>
       </simpleType>
       <simpleType>
         <restriction base="dns:ip4AddressType"/>
       </simpleType>
       <simpleType>
         <restriction base="dns:ip6AddressType"/>
       </simpleType>
     </union>
   </simpleType>

   <!-- - - - - - - - - - - - - - - - - - - - -->
   <!-- TYPE element [RFC3597] (5)        -->
   <!-- - - - - - - - - - - - - - - - - - - - -->

   <element name="TYPE" substitutionGroup="dns:RR" type="dns:TYPEType"/
>

   <complexType name="TYPEType">
     <simpleContent>
       <extension base="dns:hexWithWhitespace">
         <attributeGroup ref="dns:baseAttributes"/>
         <attribute name="rrtype" type="unsignedShort" use="required"/>
       </extension>
     </simpleContent>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - -->
   <!-- A element [RFC1035] (3.4.1)       -->
```

```
   <!-- - - - - - - - - - - - - - - - - - - - -->

   <element name="A" substitutionGroup="dns:RR" type="dns:AType"/>

   <complexType name="AType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="address" type="dns:ip4AddressType"
use="required"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - -->
   <!-- A6 element [RFC2874] (3.1.3)      -->
   <!-- - - - - - - - - - - - - - - - - - - - -->

   <element name="A6" substitutionGroup="dns:RR" type="dns:A6Type"/>

   <complexType name="A6Type">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="prefixlength" type="dns:prefixLengthType"
       use="required"/>
     <attribute name="addresssuffix" type="dns:ip6AddressType"
       use="optional"/>
     <attribute name="prefixname" type="dns:domainType" use="optional"/
>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - -->
   <!-- AAAA element [RFC3596] (2.2)      -->
   <!-- - - - - - - - - - - - - - - - - - - - -->

   <element name="AAAA" substitutionGroup="dns:RR" type="dns:AAAAType"/
>

   <complexType name="AAAAType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="ip6address" type="dns:ip6AddressType"
       use="required"/>
   </complexType>

   <!-- - - - - - - - - - - - - - - - - - - - -->
   <!-- AFSDB element [RFC1183] (1)       -->
   <!-- - - - - - - - - - - - - - - - - - - - -->

   <element name="AFSDB" substitutionGroup="dns:RR"
            type="dns:AFSDBType"/>

   <complexType name="AFSDBType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="subtype" type="unsignedShort" use="required"/>
```

```
    <attribute name="hostname" type="dns:domainType" use="required"/>
  </complexType>

  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- APL element [RFC3123] (4)             -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="APL" substitutionGroup="dns:RR" type="dns:APLType"
    nillable="true"/>

  <complexType name="APLType">
    <simpleContent>
      <extension base="string">
        <attributeGroup ref="dns:baseAttributes"/>
        <attribute name="addressfamily" type="unsignedShort"
          use="optional"/>
        <attribute name="prefix" type="dns:prefixLengthType"
          use="optional"/>
        <attribute name="n" use="optional">
          <simpleType>
            <restriction base="string">
              <enumeration value="!"/>
            </restriction>
          </simpleType>
        </attribute>
        <attribute name="afdlength" use="optional">
          <simpleType>
            <restriction base="unsignedByte">
              <maxInclusive value="128"/>
            </restriction>
          </simpleType>
        </attribute>
        <attribute name="afdpart" type="dns:ipAddressType"
          use="optional"/>
      </extension>
    </simpleContent>
  </complexType>

  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- CERT element [RFC2538] (2.2)          -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="CERT" substitutionGroup="dns:RR" type="dns:CERTType"/
>

  <complexType name="CERTType">
    <simpleContent>
      <extension base="base64Binary">
        <attributeGroup ref="dns:baseAttributes"/>
```

```
        <attribute name="type" type="dns:certificateTypeType"
          use="required"/>
        <attribute name="keytag" type="dns:keytagType" use="required"/
>
        <attribute name="algorithm" type="dns:dnssecAlgorithmType"
          use="required"/>
      </extension>
    </simpleContent>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - - -->
  <!-- CNAME element [RFC1035] (3.3.1)   -->
  <!-- - - - - - - - - - - - - - - - - - - - - -->

  <element name="CNAME" substitutionGroup="dns:RR"
    type="dns:CNAMEType"/>

  <complexType name="CNAMEType">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="host" type="dns:domainType" use="required"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - - -->
  <!-- DHCID element [RFC4701] (3.2)     -->
  <!-- - - - - - - - - - - - - - - - - - - - - -->

  <element name="DHCID" substitutionGroup="dns:RR"
    type="dns:DHCIDType"/>

  <complexType name="DHCIDType">
    <simpleContent>
      <extension base="base64Binary">
        <attributeGroup ref="dns:baseAttributes"/>
      </extension>
    </simpleContent>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - - -->
  <!-- DLV element [RFC4431] (2)         -->
  <!-- - - - - - - - - - - - - - - - - - - - - -->

  <element name="DLV" substitutionGroup="dns:RR" type="dns:DLVType"/>

  <complexType name="DLVType">
    <simpleContent>
      <extension base="dns:hexWithWhitespace">
```

```
        <attributeGroup ref="dns:baseAttributes"/>
        <attribute name="keytag" type="dns:keytagType" use="required"/
>
        <attribute name="algorithm" type="dns:dnssecAlgorithmType"
          use="required"/>
        <attribute name="digesttype" type="dns:dnssecAlgorithmType"
          use="required"/>
      </extension>
    </simpleContent>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- DNAME element [RFC2672] (3)       -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="DNAME" substitutionGroup="dns:RR"
           type="dns:DNAMEType"/>

  <complexType name="DNAMEType">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="target" type="dns:domainType" use="required"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- DNSKEY element [RFC4034] (2.2)    -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="DNSKEY" substitutionGroup="dns:RR"
    type="dns:DNSKEYType"/>

  <complexType name="DNSKEYType">
    <simpleContent>
      <extension base="base64Binary">
        <attributeGroup ref="dns:baseAttributes"/>
        <attribute name="flags" type="unsignedShort" use="required"/>
        <attribute name="protocol" type="unsignedByte" use="required"/
>
        <attribute name="algorithm" type="dns:dnssecAlgorithmType"
          use="required"/>
      </extension>
    </simpleContent>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- DS element [RFC4034] (5.3)        -->
  <!-- - - - - - - - - - - - - - - - - - - - -->
```

```
   <element name="DS" substitutionGroup="dns:RR" type="dns:DSType"/>

   <complexType name="DSType">
     <simpleContent>
       <extension base="dns:hexWithWhitespace">
         <attributeGroup ref="dns:baseAttributes"/>
         <attribute name="keytag" type="dns:keytagType" use="required"/
>
         <attribute name="algorithm" type="dns:dnssecAlgorithmType"
           use="required"/>
         <attribute name="digesttype" type="dns:dnssecAlgorithmType"
           use="required"/>
       </extension>
     </simpleContent>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- GPOS element [RFC1712] (3)        -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="GPOS" substitutionGroup="dns:RR" type="dns:GPOSType"/
>

   <complexType name="GPOSType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="longitude" type="dns:longitudeType"
       use="required"/>
     <attribute name="latitude" type="dns:latitudeType" use="required"/
>
     <attribute name="altitude" type="dns:altitudeType" use="required"/
>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- HINFO element [RFC1035] (3.3.2)   -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="HINFO" substitutionGroup="dns:RR"
     type="dns:HINFOType"/>

   <complexType name="HINFOType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="cpu" type="string" use="required"/>
     <attribute name="os" type="string" use="required"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- IPSECKEY element [RFC4025] (3.1)  -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->
```

```
   <element name="IPSECKEY" substitutionGroup="dns:RR"
     type="dns:IPSECKEYType"/>

   <complexType name="IPSECKEYType">
     <simpleContent>
       <extension base="base64Binary">
         <attributeGroup ref="dns:baseAttributes"/>
         <attribute name="precedence" type="dns:preferenceType"
           use="required"/>
         <attribute name="gatewaytype" type="unsignedByte"
           use="required"/>
         <attribute name="gateway" type="dns:gatewayType"
use="required"
         />
       </extension>
     </simpleContent>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- ISDN element [RFC1183] (3.2)      -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="ISDN" substitutionGroup="dns:RR" type="dns:ISDNType"/
>

   <complexType name="ISDNType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="isdn-address" type="string" use="required"/>
     <!-- should this have a defined type? -->
     <attribute name="sa" type="string" use="optional"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- KEY element [RFC3445] (3)         -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="KEY" substitutionGroup="dns:RR" type="dns:KEYType"/>

   <complexType name="KEYType">
     <simpleContent>
       <extension base="base64Binary">
         <attributeGroup ref="dns:baseAttributes"/>
         <attribute name="flags" type="unsignedShort" use="required"/>
         <attribute name="protocol" type="unsignedByte" use="required"/
>
         <attribute name="algorithm" type="dns:dnssecAlgorithmType"
           use="required"/>
       </extension>
     </simpleContent>
```

```
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
   <!-- KX element [RFC2230] (3.1)                               -->
   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->

   <element name="KX" substitutionGroup="dns:RR" type="dns:KXType"/>

   <complexType name="KXType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="preference" type="dns:preferenceType"
       use="required"/>
     <attribute name="exchanger" type="dns:domainType" use="required"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
   <!-- LOC element [RFC1876] (3)                                -->
   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->

   <element name="LOC" substitutionGroup="dns:RR" type="dns:LOCType"/>

   <complexType name="LOCType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="version" type="unsignedByte"/>
     <attribute name="size" type="string" use="optional"/>
     <attribute name="horizpre" type="string" use="optional"/>
     <attribute name="vertpre" type="string" use="optional"/>
     <attribute name="latitude" type="dns:latitudeType" use="required"/
>
     <attribute name="longitude" type="dns:longitudeType"
       use="required"/>
     <attribute name="altitude" type="dns:altitudeType" use="required"/
>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
   <!-- MB element [RFC1035] (3.3.3)                             -->
   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->

   <element name="MB" substitutionGroup="dns:RR" type="dns:MBType"/>

   <complexType name="MBType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="madname" type="dns:domainType" use="required"/>
   </complexType>
```

```
   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- MG element [RFC1035] (3.3.6)      -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="MG" substitutionGroup="dns:RR" type="dns:MGType"/>

   <complexType name="MGType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="mgmname" type="dns:domainType" use="required"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- MINFO element [RFC1035] (3.3.7)   -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="MINFO" substitutionGroup="dns:RR"
            type="dns:MINFOType"/>

   <complexType name="MINFOType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="rmailbx" type="dns:domainType" use="required"/>
     <attribute name="emailbx" type="dns:domainType" use="required"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- MR element [RFC1035] (3.3.8)      -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="MR" substitutionGroup="dns:RR" type="dns:MRType"/>

   <complexType name="MRType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="newname" type="dns:domainType" use="required"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- MX element [RFC1035] (3.3.9)      -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="MX" substitutionGroup="dns:RR" type="dns:MXType"/>

   <complexType name="MXType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="preference" type="dns:preferenceType"
       use="required"/>
```

```
    <attribute name="exchange" type="dns:domainType" use="required"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- NAPTR element [RFC3403] (4.1)     -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="NAPTR" substitutionGroup="dns:RR"
           type="dns:NAPTRType"/>

  <complexType name="NAPTRType">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="order" type="unsignedInt" use="required"/>
    <attribute name="preference" type="dns:preferenceType"
      use="required"/>
    <attribute name="flags" type="string" use="required"/>
    <attribute name="services" type="string" use="required"/>
    <attribute name="regexp" type="string" use="required"/>
    <attribute name="replacement" type="dns:domainType"
      use="required"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- NS element [RFC1035] (3.3.11)     -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="NS" substitutionGroup="dns:RR" type="dns:NSType"/>

  <complexType name="NSType">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="nsdname" type="dns:domainType" use="required"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- NSAP element [RFC1706] (5)        -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="NSAP" substitutionGroup="dns:RR" type="dns:NSAPType"/
>

  <complexType name="NSAPType">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="nsap" type="string" use="required"/>
    <!-- defined type? -->
  </complexType>
```

```
   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
   <!-- NSEC element [RFC4034] (4.2)      -->
   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->

   <element name="NSEC" substitutionGroup="dns:RR" type="dns:NSECType"/
>

   <complexType name="NSECType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="nextdomainname" type="dns:domainType"
       use="required"/>
     <attribute name="typebitmaps" type="dns:rrListType"
                use="required"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
   <!-- NSEC3 element [RFC5155] (3.2)     -->
   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->

   <element name="NSEC3" substitutionGroup="dns:RR"
            type="dns:NSEC3Type"/>

   <complexType name="NSEC3Type">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="hashalgorithm" type="dns:dnssecAlgorithmType"
       use="required"/>
     <attribute name="flags" type="unsignedByte" use="required"/>
     <attribute name="iterations" type="unsignedShort" use="required"/>
     <attribute name="saltlength" type="dns:rdataLength8Type"
       use="optional"/>
     <attribute name="salt" type="hexBinary" use="required"/>
     <attribute name="hashlength" type="dns:rdataLength8Type"
       use="optional"/>
     <attribute name="nexthashedownername"
       type="dns:base32HexRestricted" use="required"/>
     <attribute name="typebitmaps" type="dns:rrListType"
       use="required"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
   <!-- NSEC3PARAM element [RFC5155] (4.2)-->
   <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->

   <element name="NSEC3PARAM" substitutionGroup="dns:RR"
     type="dns:NSEC3PARAMType"/>

   <complexType name="NSEC3PARAMType">
```

```
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="hashalgorithm" type="dns:dnssecAlgorithmType"
      use="required"/>
    <attribute name="flags" type="unsignedByte" use="required"/>
    <attribute name="iterations" type="unsignedShort" use="required"/>
    <attribute name="saltlength" type="dns:rdataLength8Type"
      use="optional"/>
    <attribute name="salt" type="hexBinary" use="required"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- NULL element [RFC1035] (3.3.10)   -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="NULL" substitutionGroup="dns:RR" type="dns:NULLType"
    nillable="true"/>

  <complexType name="NULLType" mixed="true">
    <simpleContent>
      <extension base="string">
        <attributeGroup ref="dns:baseAttributes"/>
      </extension>
    </simpleContent>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- PTR element [RFC1035] (3.3.12)    -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="PTR" substitutionGroup="dns:RR" type="dns:PTRType"/>

  <complexType name="PTRType">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="ptrdname" type="dns:domainType" use="required"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- PX element [RFC2163] (4)          -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="PX" substitutionGroup="dns:RR" type="dns:PXType"/>

  <complexType name="PXType">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="preference" type="dns:preferenceType"
```

```
      use="required"/>
    <attribute name="map822" type="dns:domainType" use="required"/>
    <attribute name="mapx400" type="dns:domainType" use="required"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - - -->
  <!-- RP element [RFC1183] (2.2)              -->
  <!-- - - - - - - - - - - - - - - - - - - - - -->

  <element name="RP" substitutionGroup="dns:RR" type="dns:RPType"/>

  <complexType name="RPType">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="mbox-dname" type="dns:domainType" use="required"/
>
    <attribute name="txt-dname" type="dns:domainType" use="required"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - - -->
  <!-- RRSIG element [RFC4034] (3.2)           -->
  <!-- - - - - - - - - - - - - - - - - - - - - -->

  <element name="RRSIG" substitutionGroup="dns:RR"
    type="dns:RRSIGType"/>

  <complexType name="RRSIGType">
    <simpleContent>
      <extension base="base64Binary">
        <attributeGroup ref="dns:baseAttributes"/>
        <attribute name="typecovered" type="dns:rrTypeType"
          use="required"/>
        <attribute name="algorithm" type="dns:dnssecAlgorithmType"
          use="required"/>
        <attribute name="labels" type="unsignedByte" use="required"/>
        <attribute name="originalttl" type="dns:ttlType"
          use="required"/>
        <attribute name="signatureexpiration"
          type="dns:secondsSinceEpochTextType" use="required"/>
        <attribute name="signatureinception"
          type="dns:secondsSinceEpochTextType" use="required"/>
        <attribute name="keytag" type="dns:keytagType" use="required"/
>
        <attribute name="signersname" type="dns:domainType"
          use="required"/>
      </extension>
    </simpleContent>
  </complexType>
```

```
   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- RT element [RFC1183] (3.3)              -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="RT" substitutionGroup="dns:RR" type="dns:RTType"/>

   <complexType name="RTType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="preference" type="dns:preferenceType"
       use="required"/>
     <attribute name="intermediate-host" type="dns:domainType"
       use="required"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- SIG element [RFC2931] & [RFC4034] -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="SIG" substitutionGroup="dns:RR" type="dns:SIGType"/>

   <complexType name="SIGType">
     <simpleContent>
       <extension base="base64Binary">
         <attributeGroup ref="dns:baseAttributes"/>
         <attribute name="typecovered" type="unsignedShort"
           use="required"/>
         <attribute name="algorithm" type="dns:dnssecAlgorithmType"
           use="required"/>
         <attribute name="labels" type="unsignedByte" use="required"/>
         <attribute name="originalttl" type="dns:ttlType"
           use="required"/>
         <attribute name="signatureexpiration"
           type="dns:secondsSinceEpochTextType" use="required"/>
         <attribute name="signatureinception"
           type="dns:secondsSinceEpochTextType" use="required"/>
         <attribute name="keytag" type="dns:keytagType" use="required"/
>
         <attribute name="signersname" type="dns:domainType"
           use="required"/>
       </extension>
     </simpleContent>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- SSHFP element [RFC4255] (3.2)     -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->
```

```
   <element name="SSHFP" substitutionGroup="dns:RR"
     type="dns:SSHFPType"/>

   <complexType name="SSHFPType">
     <simpleContent>
       <extension base="dns:hexWithWhitespace">
         <attributeGroup ref="dns:baseAttributes"/>
         <attribute name="algorithm" type="dns:sshAlgorithmType"
           use="required"/>
         <attribute name="fptype" type="dns:sshDigestType"
           use="required"/>
       </extension>
     </simpleContent>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- SOA element [RFC1035] (3.3.13)    -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="SOA" substitutionGroup="dns:RR" type="dns:SOAType"/>

   <complexType name="SOAType">
     <attributeGroup ref="dns:baseAttributes"/>
     <attribute name="mname" type="dns:domainType" use="required"/>
     <attribute name="rname" type="dns:domainType" use="required"/>
     <attribute name="serial" type="dns:serialType" use="required"/>
     <attribute name="refresh" type="dns:secondsInterval32Type"
       use="required"/>
     <attribute name="retry" type="dns:secondsInterval32Type"
       use="required"/>
     <attribute name="expire" type="dns:secondsInterval32Type"
       use="required"/>
     <attribute name="minimum" type="dns:ttlType" use="required"/>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - - - - -->
   <!-- SPF element [RFC4408] (3.1.1)     -->
   <!-- - - - - - - - - - - - - - - - - - - - - -->

   <element name="SPF" substitutionGroup="dns:RR" type="dns:SPFType"/>

   <complexType name="SPFType">
     <simpleContent>
       <extension base="string">
         <attributeGroup ref="dns:baseAttributes"/>
       </extension>
```

```
    </simpleContent>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- SRV element [RFC2782] (0)         -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="SRV" substitutionGroup="dns:RR" type="dns:SRVType"/>

  <complexType name="SRVType">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="priority" type="unsignedShort" use="required"/>
    <attribute name="weight" type="unsignedShort" use="required"/>
    <attribute name="port" type="dns:portType" use="required"/>
    <attribute name="target" type="dns:domainType" use="required"/>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- TKEY element [RFC2930] (2)        -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="TKEY" substitutionGroup="dns:RR" type="dns:TKEYType"
    nillable="true"/>

  <complexType name="TKEYType">
    <simpleContent>
      <extension base="dns:hexWithWhitespace">
        <attributeGroup ref="dns:baseAttributes"/>
        <attribute name="algorithm" type="dns:domainType"
          use="required"/>
        <attribute name="inception" type="dns:secondsSinceEpoch32Type"
          use="required"/>
        <attribute name="expiration"
type="dns:secondsSinceEpoch32Type"
          use="required"/>
        <attribute name="mode" type="unsignedShort" use="required"/>
        <attribute name="error" type="dns:rcode16Type" use="required"/
>
        <attribute name="keysize" type="unsignedShort" use="required"/
>
        <attribute name="keydata" type="hexBinary" use="required"/>
        <attribute name="othersize" type="dns:rdataLengthType"
          use="optional"/>
      </extension>
    </simpleContent>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
```

```
   <!-- TSIG element [RFC2845] (2.3)      -->
   <!-- - - - - - - - - - - - - - - - - - -->

   <element name="TSIG" substitutionGroup="dns:RR" type="dns:TSIGType"
     nillable="true"/>

   <complexType name="TSIGType">
     <simpleContent>
       <extension base="dns:hexWithWhitespace">
         <attributeGroup ref="dns:baseAttributes"/>
         <attribute name="algorithm" type="dns:domainType"
           use="required"/>
         <attribute name="timesigned"
type="dns:secondsSinceEpoch48Type"
           use="required"/>
         <attribute name="fudge" type="unsignedShort" use="required"/>
         <attribute name="macsize" type="unsignedShort" use="required"/
>
         <attribute name="mac" type="hexBinary" use="required"/>
         <attribute name="originalid" type="dns:idType" use="required"/
>
         <attribute name="error" type="dns:rcode16Type" use="optional"/
>
         <attribute name="otherlen" type="dns:rdataLengthType"
           use="optional"/>
       </extension>
     </simpleContent>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - -->
   <!-- TXT element [RFC1035] (3.3.14)    -->
   <!-- - - - - - - - - - - - - - - - - - -->

   <element name="TXT" substitutionGroup="dns:RR" type="dns:TXTType"/>

   <complexType name="TXTType">
     <simpleContent>
       <extension base="string">
         <attributeGroup ref="dns:baseAttributes"/>
       </extension>
     </simpleContent>
   </complexType>


   <!-- - - - - - - - - - - - - - - - - - -->
   <!-- WKS element [RFC1035] (3.4.2)     -->
   <!-- - - - - - - - - - - - - - - - - - -->

   <element name="WKS" substitutionGroup="dns:RR" type="dns:WKSType"/>

   <complexType name="WKSType">
```

```
    <simpleContent>
      <extension base="dns:hexWithWhitespace">
        <attributeGroup ref="dns:baseAttributes"/>
        <attribute name="address" type="dns:ip4AddressType"
          use="required"/>
        <attribute name="protocol" type="dns:ipProtocolType"
          use="required"/>
      </extension>
    </simpleContent>
  </complexType>


  <!-- - - - - - - - - - - - - - - - - - - - -->
  <!-- X25 element [RFC1183] (3.1)       -->
  <!-- - - - - - - - - - - - - - - - - - - - -->

  <element name="X25" substitutionGroup="dns:RR" type="dns:X25Type"/>

  <complexType name="X25Type">
    <attributeGroup ref="dns:baseAttributes"/>
    <attribute name="psdn-address" type="string" use="required"/>
  </complexType>

</schema>
```

5. Acknowledgements

We would like to thank Alex Dalitz and Roy Arends for their review of early versions of this draft.

The regular expression for IPv6 addresses was published by Dartware and altered by the authors to fit with the limited regular expression syntax of XML Schema.

6. IANA Considerations

This memo uses URNs to describe XML namespaces [W3C.REC-xml-names-20091208] and XML schemas conforming to a registry mechanism described in [RFC3688]. Three URI assignments need to be registered by the IANA.

Registration request for the dnsxml namespace:

URI: urn:ietf:params:xml:ns:dns

Registrant Contact: See the "Author's Address" section of this memo.

XML: None. Namespace URIs do not represent an XML specification.

Registration request for the dnsxml version specific namespace:

URI: urn:ietf:params:xml:ns:dns-1.0

Registrant Contact: See the "Author's Address" section of this memo.

XML: None. Namespace URIs do not represent an XML specification.

Registration request for the dnsxml XML schema:

URI: urn:ietf:params:xml:schema:dns-1.0

Registrant Contact: See the "Author's Address" section of this memo.

XML: See Section 4 of this memo.

7. Security Considerations

This memo includes no security considerations.

8. References

8.1. Normative References

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC1183] Everhart, C., Mamakos, L., Ullmann, R., and P. Mockapetris, "New DNS RR Definitions", RFC 1183, October 1990.

[RFC1706] Manning, B. and R. Colella, "DNS NSAP Resource Records", RFC 1706, October 1994.

[RFC1712] Farrell, C., Schulze, M., Pleitner, S., and D. Baldoni, "DNS Encoding of Geographical Location", RFC 1712, November 1994.

[RFC1876] Davis, C., Vixie, P., Goodwin, T., and I. Dickinson, "A Means for Expressing Location Information in the Domain Name System", RFC 1876, January 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.

[RFC2163] Allocchio, C., "Using the Internet DNS to Distribute MIXER Conformant Global Address Mapping (MCGAM)", RFC 2163, January 1998.

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.

[RFC2230] Atkinson, R., "Key Exchange Delegation Record for the DNS", RFC 2230, November 1997.

[RFC2538] Eastlake, D. and O. Gudmundsson, "Storing Certificates in the Domain Name System (DNS)", RFC 2538, March 1999.

[RFC2672] Crawford, M., "Non-Terminal DNS Name Redirection", RFC 2672, August 1999.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, May 2000.

[RFC2874] Crawford, M. and C. Huitema, "DNS Extensions to Support IPv6 Address Aggregation and Renumbering", RFC 2874, July 2000.

[RFC2930] Eastlake, D., "Secret Key Establishment for DNS (TKEY RR)", RFC 2930, September 2000.

[RFC2931] Eastlake, D., "DNS Request and Transaction Signatures (SIG(0)s)", RFC 2931, September 2000.

[RFC3123] Koch, P., "A DNS RR Type for Lists of Address Prefixes (APL RR)", RFC 3123, June 2001.

[RFC3403] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database", RFC 3403, October 2002.

[RFC3445] Massey, D. and S. Rose, "Limiting the Scope of the KEY Resource Record (RR)", RFC 3445, December 2002.

[RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", RFC 3596, October 2003.

[RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record (RR) Types", RFC 3597, September 2003.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC4025] Richardson, M., "A Method for Storing IPsec Keying Material in DNS", RFC 4025, March 2005.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.

[RFC4255] Schlyter, J. and W. Griffin, "Using DNS to Securely Publish Secure Shell (SSH) Key Fingerprints", RFC 4255, January 2006.

[RFC4408] Wong, M. and W. Schlitt, "Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1", RFC 4408, April 2006.

[RFC4431] Andrews, M. and S. Weiler, "The DNSSEC Lookaside Validation (DLV) DNS Resource Record", RFC 4431, February 2006.

[RFC4509] Hardaker, W., "Use of SHA-256 in DNSSEC Delegation Signer (DS) Resource Records (RRs)", RFC 4509, May 2006.

[RFC4701] Stapp, M., Lemon, T., and A. Gustafsson, "A DNS Resource Record (RR) for Encoding Dynamic Host Configuration Protocol (DHCP) Information (DHCID RR)", RFC 4701, October 2006.

[RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, March 2008.

[RFC5702] Jansen, J., "Use of SHA-2 Algorithms with RSA in DNSKEY and RRSIG Resource Records for DNSSEC", RFC 5702, October 2009.

[RFC5933] Dolmatov, V., Chuprina, A., and I. Ustinov, "Use of GOST Signature Algorithms in DNSKEY and RRSIG Resource Records for DNSSEC", RFC 5933, July 2010.

[RFC6195] Eastlake, D., "Domain Name System (DNS) IANA Considerations", RFC 6195, March 2011.

[RFC6605] Hoffman, P. and W. Wijngaards, "Elliptic Curve Digital Signature Algorithm (DSA) for DNSSEC", RFC 6605, April 2012.

[W3C.REC-xml-20081126] Yergeau, F., Maler, E., Paoli, J., Sperberg-McQueen, C., and T. Bray, "Extensible Markup Language (XML) 1.0 (Fifth Edition)", World Wide Web Consortium Recommendation REC-xml-20081126, November 2008, <http://www.w3.org/TR/2008/REC-xml-20081126>.

[W3C.REC-xml-names-20091208] Hollander, D., Layman, A., Bray, T., Tobin, R., and H. Thompson, "Namespaces in XML 1.0 (Third Edition)", World Wide Web Consortium Recommendation REC-xml-names-20091208, December 2009, <http://www.w3.org/TR/2009/REC-xml-names-20091208>.

[W3C.REC-xmlschema-1-20041028] Beech, D., Thompson, H., Maloney, M., and N. Mendelsohn, "XML Schema Part 1: Structures Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-1-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028>.

[W3C.REC-xmlschema-2-20041028] Biron, P. and A. Malhotra, "XML Schema Part 2: Datatypes Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-2-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028>.

[dns-sec-alg-numbers] IANA, "Domain Name System Security (DNSSEC) Algorithm Numbers", 2012, <http://www.iana.org/assignments/dns-sec-alg-numbers/dns-sec-alg-numbers.xml>.

8.2. Informative References

[RFC1002] NetBIOS Working Group, "Protocol standard for a NetBIOS service on a TCP/UDP transport: Detailed specifications", STD 19, RFC 1002, March 1987.

[RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[RFC2671] Vixie, P., "Extension Mechanisms for DNS (EDNS0)", RFC 2671, August 1999.

Authors' Addresses

Jay Daley (editor)
.nz Registry Services
PO Box 24361, Manners Street
Wellington 6142
New Zealand

Phone: +64 4 931 6970
Email: jay@nzrs.net.nz

Stephen Morris
Internet Systems Consortium
Grove
UK

John Dickinson
Sinodun
Wallingford
UK