                    Clarifications to the DNS Specification

Status of this Memo

   This document specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements. Please refer to the current edition of the "Internet
   Official Protocol Standards" (STD 1) for the standardization state
   and status of this protocol. Distribution of this memo is unlimited.

1. Abstract

   This document considers some areas that have been identified as
   problems with the specification of the Domain Name System, and
   proposes remedies for the defects identified. Eight separate issues
   are considered:

      + IP packet header address usage from multi-homed servers,
      + TTLs in sets of records with the same name, class, and type,
      + correct handling of zone cuts,
      + three minor issues concerning SOA records and their use,
      + the precise definition of the Time to Live (TTL)
      + Use of the TC (truncated) header bit
      + the issue of what is an authoritative, or canonical, name,
      + and the issue of what makes a valid DNS label.

   The first six of these are areas where the correct behaviour has been
   somewhat unclear, we seek to rectify that. The other two are already
   adequately specified, however the specifications seem to be sometimes
   ignored. We seek to reinforce the existing specifications.

Contents

2. Introduction

   Several problem areas in the Domain Name System specification
   [RFC1034, RFC1035] have been noted through the years [RFC1123]. This
   document addresses several additional problem areas. The issues here
   are independent. Those issues are the question of which source
   address a multi-homed DNS server should use when replying to a query,
   the issue of differing TTLs for DNS records with the same label,
   class and type, and the issue of canonical names, what they are, how
   CNAME records relate, what names are legal in what parts of the DNS,
   and what is the valid syntax of a DNS name.

   Clarifications to the DNS specification to avoid these problems are
   made in this memo. A minor ambiguity in RFC1034 concerned with SOA
   records is also corrected, as is one in the definition of the TTL
   (Time To Live) and some possible confusion in use of the TC bit.

3. Terminology

   This memo does not use the oft used expressions MUST, SHOULD, MAY, or
   their negative forms. In some sections it may seem that a
   specification is worded mildly, and hence some may infer that the
   specification is optional. That is not correct. Anywhere that this
   memo suggests that some action should be carried out, or must be
   carried out, or that some behaviour is acceptable, or not, that is to
   be considered as a fundamental aspect of this specification,
   regardless of the specific words used. If some behaviour or action
   is truly optional, that will be clearly specified by the text.

4. Server Reply Source Address Selection

   Most, if not all, DNS clients, expect the address from which a reply
   is received to be the same address as that to which the query
   eliciting the reply was sent. This is true for servers acting as
   clients for the purposes of recursive query resolution, as well as
   simple resolver clients. The address, along with the identifier (ID)
   in the reply is used for disambiguating replies, and filtering
   spurious responses. This may, or may not, have been intended when
   the DNS was designed, but is now a fact of life.

   Some multi-homed hosts running DNS servers generate a reply using a
   source address that is not the same as the destination address from
   the client's request packet. Such replies will be discarded by the
   client because the source address of the reply does not match that of
   a host to which the client sent the original request. That is, it
   appears to be an unsolicited response.

4.1. UDP Source Address Selection

   To avoid these problems, servers when responding to queries using UDP
   must cause the reply to be sent with the source address field in the
   IP header set to the address that was in the destination address
   field of the IP header of the packet containing the query causing the
   response. If this would cause the response to be sent from an IP
   address that is not permitted for this purpose, then the response may
   be sent from any legal IP address allocated to the server. That
   address should be chosen to maximise the possibility that the client
   will be able to use it for further queries. Servers configured in
   such a way that not all their addresses are equally reachable from
   all potential clients need take particular care when responding to
   queries sent to anycast, multicast, or similar, addresses.

4.2. Port Number Selection

   Replies to all queries must be directed to the port from which they
   were sent. When queries are received via TCP this is an inherent
   part of the transport protocol. For queries received by UDP the
   server must take note of the source port and use that as the
   destination port in the response. Replies should always be sent from
   the port to which they were directed. Except in extraordinary
   circumstances, this will be the well known port assigned for DNS
   queries [RFC1700].

5. Resource Record Sets

   Each DNS Resource Record (RR) has a label, class, type, and data. It
   is meaningless for two records to ever have label, class, type and
   data all equal - servers should suppress such duplicates if
   encountered. It is however possible for most record types to exist
   with the same label, class and type, but with different data. Such a
   group of records is hereby defined to be a Resource Record Set
   (RRSet).

5.1. Sending RRs from an RRSet

   A query for a specific (or non-specific) label, class, and type, will
   always return all records in the associated RRSet - whether that be
   one or more RRs. The response must be marked as "truncated" if the
   entire RRSet will not fit in the response.

5.2. TTLs of RRs in an RRSet

   Resource Records also have a time to live (TTL). It is possible for
   the RRs in an RRSet to have different TTLs. No uses for this have
   been found that cannot be better accomplished in other ways. This
   can, however, cause partial replies (not marked "truncated") from a
   caching server, where the TTLs for some but not all the RRs in the
   RRSet have expired.

   Consequently the use of differing TTLs in an RRSet is hereby
   deprecated, the TTLs of all RRs in an RRSet must be the same.

   Should a client receive a response containing RRs from an RRSet with
   differing TTLs, it should treat this as an error. If the RRSet
   concerned is from a non-authoritative source for this data, the
   client should simply ignore the RRSet, and if the values were
   required, seek to acquire them from an authoritative source. Clients
   that are configured to send all queries to one, or more, particular
   servers should treat those servers as authoritative for this purpose.
   Should an authoritative source send such a malformed RRSet, the

client should treat the RRs for all purposes as if all TTLs in the
RRSet had been set to the value of the lowest TTL in the RRSet. In
no case may a server send an RRSet with TTLs not all equal.

5.3. DNSSEC Special Cases

Two of the record types added by DNS Security (DNSSEC) [RFC2065]
require special attention when considering the formation of Resource
Record Sets. Those are the SIG and NXT records. It should be noted
that DNS Security is still very new, and there is, as yet, little
experience with it. Readers should be prepared for the information
related to DNSSEC contained in this document to become outdated as
the DNS Security specification matures.

5.3.1. SIG records and RRSets

A SIG record provides signature (validation) data for another RRSet
in the DNS. Where a zone has been signed, every RRSet in the zone
will have had a SIG record associated with it. The data type of the
RRSet is included in the data of the SIG RR, to indicate with which
particular RRSet this SIG record is associated. Were the rules above
applied, whenever a SIG record was included with a response to
validate that response, the SIG records for all other RRSets
associated with the appropriate node would also need to be included.
In some cases, this could be a very large number of records, not
helped by their being rather large RRs.

Thus, it is specifically permitted for the authority section to
contain only those SIG RRs with the "type covered" field equal to the
type field of an answer being returned. However, where SIG records
are being returned in the answer section, in response to a query for
SIG records, or a query for all records associated with a name
(type=ANY) the entire SIG RRSet must be included, as for any other RR
type.

Servers that receive responses containing SIG records in the
authority section, or (probably incorrectly) as additional data, must
understand that the entire RRSet has almost certainly not been
included. Thus, they must not cache that SIG record in a way that
would permit it to be returned should a query for SIG records be
received at that server. RFC2065 actually requires that SIG queries
be directed only to authoritative servers to avoid the problems that
could be caused here, and while servers exist that do not understand
the special properties of SIG records, this will remain necessary.
However, careful design of SIG record processing in new
implementations should permit this restriction to be relaxed in the
future, so resolvers do not need to treat SIG record queries
specially.

It has been occasionally stated that a received request for a SIG
record should be forwarded to an authoritative server, rather than
being answered from data in the cache. This is not necessary - a
server that has the knowledge of SIG as a special case for processing
this way would be better to correctly cache SIG records, taking into
account their characteristics. Then the server can determine when it
is safe to reply from the cache, and when the answer is not available
and the query must be forwarded.

5.3.2. NXT RRs

Next Resource Records (NXT) are even more peculiar. There will only
ever be one NXT record in a zone for a particular label, so
superficially, the RRSet problem is trivial. However, at a zone cut,
both the parent zone, and the child zone (superzone and subzone in
RFC2065 terminology) will have NXT records for the same name. Those
two NXT records do not form an RRSet, even where both zones are
housed at the same server. NXT RRSets always contain just a single
RR. Where both NXT records are visible, two RRSets exist. However,
servers are not required to treat this as a special case when
receiving NXT records in a response. They may elect to notice the
existence of two different NXT RRSets, and treat that as they would
two different RRSets of any other type. That is, cache one, and
ignore the other. Security aware servers will need to correctly
process the NXT record in the received response though.

5.4. Receiving RRSets

Servers must never merge RRs from a response with RRs in their cache
to form an RRSet. If a response contains data that would form an
RRSet with data in a server's cache the server must either ignore the
RRs in the response, or discard the entire RRSet currently in the
cache, as appropriate. Consequently the issue of TTLs varying
between the cache and a response does not cause concern, one will be
ignored. That is, one of the data sets is always incorrect if the
data from an answer differs from the data in the cache. The
challenge for the server is to determine which of the data sets is
correct, if one is, and retain that, while ignoring the other. Note
that if a server receives an answer containing an RRSet that is
identical to that in its cache, with the possible exception of the
TTL value, it may, optionally, update the TTL in its cache with the
TTL of the received answer. It should do this if the received answer
would be considered more authoritative (as discussed in the next
section) than the previously cached answer.

5.4.1. Ranking data

   When considering whether to accept an RRSet in a reply, or retain an
   RRSet already in its cache instead, a server should consider the
   relative likely trustworthiness of the various data. An
   authoritative answer from a reply should replace cached data that had
   been obtained from additional information in an earlier reply.
   However additional information from a reply will be ignored if the
   cache contains data from an authoritative answer or a zone file.

   The accuracy of data available is assumed from its source.
   Trustworthiness shall be, in order from most to least:

     + Data from a primary zone file, other than glue data,
     + Data from a zone transfer, other than glue,
     + The authoritative data included in the answer section of an
       authoritative reply.
     + Data from the authority section of an authoritative answer,
     + Glue from a primary zone, or glue from a zone transfer,
     + Data from the answer section of a non-authoritative answer, and
       non-authoritative data from the answer section of authoritative
       answers,
     + Additional information from an authoritative answer,
       Data from the authority section of a non-authoritative answer,
       Additional information from non-authoritative answers.

   Note that the answer section of an authoritative answer normally
   contains only authoritative data. However when the name sought is an
   alias (see section 10.1.1) only the record describing that alias is
   necessarily authoritative. Clients should assume that other records
   may have come from the server's cache. Where authoritative answers
   are required, the client should query again, using the canonical name
   associated with the alias.

   Unauthenticated RRs received and cached from the least trustworthy of
   those groupings, that is data from the additional data section, and
   data from the authority section of a non-authoritative answer, should
   not be cached in such a way that they would ever be returned as
   answers to a received query. They may be returned as additional
   information where appropriate. Ignoring this would allow the
   trustworthiness of relatively untrustworthy data to be increased
   without cause or excuse.

   When DNS security [RFC2065] is in use, and an authenticated reply has
   been received and verified, the data thus authenticated shall be
   considered more trustworthy than unauthenticated data of the same
   type. Note that throughout this document, "authoritative" means a
   reply with the AA bit set. DNSSEC uses trusted chains of SIG and KEY

records to determine the authenticity of data, the AA bit is almost
irrelevant. However DNSSEC aware servers must still correctly set
the AA bit in responses to enable correct operation with servers that
are not security aware (almost all currently).

Note that, glue excluded, it is impossible for data from two
correctly configured primary zone files, two correctly configured
secondary zones (data from zone transfers) or data from correctly
configured primary and secondary zones to ever conflict. Where glue
for the same name exists in multiple zones, and differs in value, the
nameserver should select data from a primary zone file in preference
to secondary, but otherwise may choose any single set of such data.
Choosing that which appears to come from a source nearer the
authoritative data source may make sense where that can be
determined. Choosing primary data over secondary allows the source
of incorrect glue data to be discovered more readily, when a problem
with such data exists. Where a server can detect from two zone files
that one or more are incorrectly configured, so as to create
conflicts, it should refuse to load the zones determined to be
erroneous, and issue suitable diagnostics.

"Glue" above includes any record in a zone file that is not properly
part of that zone, including nameserver records of delegated sub-
zones (NS records), address records that accompany those NS records
(A, AAAA, etc), and any other stray data that might appear.

5.5. Sending RRSets (reprise)

A Resource Record Set should only be included once in any DNS reply.
It may occur in any of the Answer, Authority, or Additional
Information sections, as required. However it should not be repeated
in the same, or any other, section, except where explicitly required
by a specification. For example, an AXFR response requires the SOA
record (always an RRSet containing a single RR) be both the first and
last record of the reply. Where duplicates are required this way,
the TTL transmitted in each case must be the same.

6. Zone Cuts

The DNS tree is divided into "zones", which are collections of
domains that are treated as a unit for certain management purposes.
Zones are delimited by "zone cuts". Each zone cut separates a
"child" zone (below the cut) from a "parent" zone (above the cut).
The domain name that appears at the top of a zone (just below the cut
that separates the zone from its parent) is called the zone's
"origin". The name of the zone is the same as the name of the domain
at the zone's origin. Each zone comprises that subset of the DNS
tree that is at or below the zone's origin, and that is above the

cuts that separate the zone from its children (if any). The
existence of a zone cut is indicated in the parent zone by the
existence of NS records specifying the origin of the child zone. A
child zone does not contain any explicit reference to its parent.

6.1. Zone authority

The authoritative servers for a zone are enumerated in the NS records
for the origin of the zone, which, along with a Start of Authority
(SOA) record are the mandatory records in every zone. Such a server
is authoritative for all resource records in a zone that are not in
another zone. The NS records that indicate a zone cut are the
property of the child zone created, as are any other records for the
origin of that child zone, or any sub-domains of it. A server for a
zone should not return authoritative answers for queries related to
names in another zone, which includes the NS, and perhaps A, records
at a zone cut, unless it also happens to be a server for the other
zone.

Other than the DNSSEC cases mentioned immediately below, servers
should ignore data other than NS records, and necessary A records to
locate the servers listed in the NS records, that may happen to be
configured in a zone at a zone cut.

6.2. DNSSEC issues

The DNS security mechanisms [RFC2065] complicate this somewhat, as
some of the new resource record types added are very unusual when
compared with other DNS RRs. In particular the NXT ("next") RR type
contains information about which names exist in a zone, and hence
which do not, and thus must necessarily relate to the zone in which
it exists. The same domain name may have different NXT records in
the parent zone and the child zone, and both are valid, and are not
an RRSet. See also section 5.3.2.

Since NXT records are intended to be automatically generated, rather
than configured by DNS operators, servers may, but are not required
to, retain all differing NXT records they receive regardless of the
rules in section 5.4.

For a secure parent zone to securely indicate that a subzone is
insecure, DNSSEC requires that a KEY RR indicating that the subzone
is insecure, and the parent zone's authenticating SIG RR(s) be
present in the parent zone, as they by definition cannot be in the
subzone. Where a subzone is secure, the KEY and SIG records will be
present, and authoritative, in that zone, but should also always be
present in the parent zone (if secure).

Note that in none of these cases should a server for the parent zone,
not also being a server for the subzone, set the AA bit in any
response for a label at a zone cut.

7. SOA RRs

Three minor issues concerning the Start of Zone of Authority (SOA)
Resource Record need some clarification.

7.1. Placement of SOA RRs in authoritative answers

RFC1034, in section 3.7, indicates that the authority section of an
authoritative answer may contain the SOA record for the zone from
which the answer was obtained. When discussing negative caching,
RFC1034 section 4.3.4 refers to this technique but mentions the
additional section of the response. The former is correct, as is
implied by the example shown in section 6.2.5 of RFC1034. SOA
records, if added, are to be placed in the authority section.

7.2. TTLs on SOA RRs

It may be observed that in section 3.2.1 of RFC1035, which defines
the format of a Resource Record, that the definition of the TTL field
contains a throw away line which states that the TTL of an SOA record
should always be sent as zero to prevent caching. This is mentioned
nowhere else, and has not generally been implemented.
Implementations should not assume that SOA records will have a TTL of
zero, nor are they required to send SOA records with a TTL of zero.

7.3. The SOA.MNAME field

It is quite clear in the specifications, yet seems to have been
widely ignored, that the MNAME field of the SOA record should contain
the name of the primary (master) server for the zone identified by
the SOA. It should not contain the name of the zone itself. That
information would be useless, as to discover it, one needs to start
with the domain name of the SOA record - that is the name of the
zone.

8. Time to Live (TTL)

The definition of values appropriate to the TTL field in STD 13 is
not as clear as it could be, with respect to how many significant
bits exist, and whether the value is signed or unsigned. It is
hereby specified that a TTL value is an unsigned number, with a
minimum value of 0, and a maximum value of 2147483647. That is, a
maximum of $2^{31} - 1$. When transmitted, this value shall be encoded
in the less significant 31 bits of the 32 bit TTL field, with the

most significant, or sign, bit set to zero.

Implementations should treat TTL values received with the most significant bit set as if the entire value received was zero.

Implementations are always free to place an upper bound on any TTL received, and treat any larger values as if they were that upper bound. The TTL specifies a maximum time to live, not a mandatory time to live.

9. The TC (truncated) header bit

The TC bit should be set in responses only when an RRSet is required as a part of the response, but could not be included in its entirety. The TC bit should not be set merely because some extra information could have been included, but there was insufficient room. This includes the results of additional section processing. In such cases the entire RRSet that will not fit in the response should be omitted, and the reply sent as is, with the TC bit clear. If the recipient of the reply needs the omitted data, it can construct a query for that data and send that separately.

Where TC is set, the partial RRSet that would not completely fit may be left in the response. When a DNS client receives a reply with TC set, it should ignore that response, and query again, using a mechanism, such as a TCP connection, that will permit larger replies.

10. Naming issues

It has sometimes been inferred from some sections of the DNS specification [RFC1034, RFC1035] that a host, or perhaps an interface of a host, is permitted exactly one authoritative, or official, name, called the canonical name. There is no such requirement in the DNS.

10.1. CNAME resource records

The DNS CNAME ("canonical name") record exists to provide the canonical name associated with an alias name. There may be only one such canonical name for any one alias. That name should generally be a name that exists elsewhere in the DNS, though there are some rare applications for aliases with the accompanying canonical name undefined in the DNS. An alias name (label of a CNAME record) may, if DNSSEC is in use, have SIG, NXT, and KEY RRs, but may have no other data. That is, for any label in the DNS (any domain name) exactly one of the following is true:

        + one CNAME record exists, optionally accompanied by SIG, NXT, and
          KEY RRs,
        + one or more records exist, none being CNAME records,
        + the name exists, but has no associated RRs of any type,
        + the name does not exist at all.

10.1.1. CNAME terminology

   It has been traditional to refer to the label of a CNAME record as "a
   CNAME". This is unfortunate, as "CNAME" is an abbreviation of
   "canonical name", and the label of a CNAME record is most certainly
   not a canonical name. It is, however, an entrenched usage. Care
   must therefore be taken to be very clear whether the label, or the
   value (the canonical name) of a CNAME resource record is intended.
   In this document, the label of a CNAME resource record will always be
   referred to as an alias.

10.2. PTR records

   Confusion about canonical names has lead to a belief that a PTR
   record should have exactly one RR in its RRSet. This is incorrect,
   the relevant section of RFC1034 (section 3.6.2) indicates that the
   value of a PTR record should be a canonical name. That is, it should
   not be an alias. There is no implication in that section that only
   one PTR record is permitted for a name. No such restriction should
   be inferred.

   Note that while the value of a PTR record must not be an alias, there
   is no requirement that the process of resolving a PTR record not
   encounter any aliases. The label that is being looked up for a PTR
   value might have a CNAME record. That is, it might be an alias. The
   value of that CNAME RR, if not another alias, which it should not be,
   will give the location where the PTR record is found. That record
   gives the result of the PTR type lookup. This final result, the
   value of the PTR RR, is the label which must not be an alias.

10.3. MX and NS records

   The domain name used as the value of a NS resource record, or part of
   the value of a MX resource record must not be an alias. Not only is
   the specification clear on this point, but using an alias in either
   of these positions neither works as well as might be hoped, nor well
   fulfills the ambition that may have led to this approach. This
   domain name must have as its value one or more address records.
   Currently those will be A records, however in the future other record
   types giving addressing information may be acceptable. It can also
   have other RRs, but never a CNAME RR.

Searching for either NS or MX records causes "additional section
processing" in which address records associated with the value of the
record sought are appended to the answer. This helps avoid needless
extra queries that are easily anticipated when the first was made.

Additional section processing does not include CNAME records, let
alone the address records that may be associated with the canonical
name derived from the alias. Thus, if an alias is used as the value
of an NS or MX record, no address will be returned with the NS or MX
value. This can cause extra queries, and extra network burden, on
every query. It is trivial for the DNS administrator to avoid this
by resolving the alias and placing the canonical name directly in the
affected record just once when it is updated or installed. In some
particular hard cases the lack of the additional section address
records in the results of a NS lookup can cause the request to fail.

11. Name syntax

Occasionally it is assumed that the Domain Name System serves only
the purpose of mapping Internet host names to data, and mapping
Internet addresses to host names. This is not correct, the DNS is a
general (if somewhat limited) hierarchical database, and can store
almost any kind of data, for almost any purpose.

The DNS itself places only one restriction on the particular labels
that can be used to identify resource records. That one restriction
relates to the length of the label and the full name. The length of
any one label is limited to between 1 and 63 octets. A full domain
name is limited to 255 octets (including the separators). The zero
length full name is defined as representing the root of the DNS tree,
and is typically written and displayed as ".". Those restrictions
aside, any binary string whatever can be used as the label of any
resource record. Similarly, any binary string can serve as the value
of any record that includes a domain name as some or all of its value
(SOA, NS, MX, PTR, CNAME, and any others that may be added).
Implementations of the DNS protocols must not place any restrictions
on the labels that can be used. In particular, DNS servers must not
refuse to serve a zone because it contains labels that might not be
acceptable to some DNS client programs. A DNS server may be
configurable to issue warnings when loading, or even to refuse to
load, a primary zone containing labels that might be considered
questionable, however this should not happen by default.

Note however, that the various applications that make use of DNS data
can have restrictions imposed on what particular values are
acceptable in their environment. For example, that any binary label
can have an MX record does not imply that any binary name can be used
as the host part of an e-mail address. Clients of the DNS can impose

whatever restrictions are appropriate to their circumstances on the
values they use as keys for DNS lookup requests, and on the values
returned by the DNS. If the client has such restrictions, it is
solely responsible for validating the data from the DNS to ensure
that it conforms before it makes any use of that data.

See also [RFC1123] section 6.1.3.5.

12. Security Considerations

This document does not consider security.

In particular, nothing in section 4 is any way related to, or useful
for, any security related purposes.

Section 5.4.1 is also not related to security. Security of DNS data
will be obtained by the Secure DNS [RFC2065], which is mostly
orthogonal to this memo.

It is not believed that anything in this document adds to any
security issues that may exist with the DNS, nor does it do anything
to that will necessarily lessen them. Correct implementation of the
clarifications in this document might play some small part in
limiting the spread of non-malicious bad data in the DNS, but only
DNSSEC can help with deliberate attempts to subvert DNS data.

13. References

[RFC1034]   Mockapetris, P., "Domain Names - Concepts and Facilities",
             STD 13, RFC 1034, November 1987.

[RFC1035]   Mockapetris, P., "Domain Names - Implementation and
             Specification", STD 13, RFC 1035, November 1987.

[RFC1123]   Braden, R., "Requirements for Internet Hosts - application
             and support", STD 3, RFC 1123, January 1989.

[RFC1700]   Reynolds, J., Postel, J., "Assigned Numbers",
             STD 2, RFC 1700, October 1994.

[RFC2065]   Eastlake, D., Kaufman, C., "Domain Name System Security
             Extensions", RFC 2065, January 1997.

14. Acknowledgements

   This memo arose from discussions in the DNSIND working group of the
   IETF in 1995 and 1996, the members of that working group are largely
   responsible for the ideas captured herein. Particular thanks to
   Donald E. Eastlake, 3rd, and Olafur Gudmundsson, for help with the
   DNSSEC issues in this document, and to John Gilmore for pointing out
   where the clarifications were not necessarily clarifying. Bob Halley
   suggested clarifying the placement of SOA records in authoritative
   answers, and provided the references. Michael Patton, as usual, and
   Mark Andrews, Alan Barrett and Stan Barber provided much assistance
   with many details. Josh Littlefield helped make sure that the
   clarifications didn't cause problems in some irritating corner cases.

15. Authors' Addresses

   Robert Elz
   Computer Science
   University of Melbourne
   Parkville, Victoria, 3052
   Australia.

   EMail: kre@munnari.OZ.AU

   Randy Bush
   RGnet, Inc.
   5147 Crystal Springs Drive NE
   Bainbridge Island, Washington, 98110
   United States.

   EMail: randy@psg.com