Network Working Group   P. Prindeville
Request for Comments: 1048   McGill University
                                                          February 1988

                 BOOTP Vendor Information Extensions

Status of this Memo

   This memo proposes an addition to the Bootstrap Protocol (BOOTP).
   Comments and suggestions for improvements are sought. Distribution
   of this memo is unlimited.

Introduction

   As workstations and personal computers proliferate on the Internet,
   the administrative complexity of maintaining a network is increased
   by an order of magnitude. The assignment of local network resources
   to each client represents one such difficulty. In most environments,
   delegating such responsibility to the user is not plausible and,
   indeed, the solution is to define the resources in uniform terms, and
   to automate their assignment.

   The basic Bootstrap Protocol [RFC-951] dealt with the issue of
   assigning an internet address to a client, as well as a few other
   resources. The protocol included provisions for vendor-defined
   resource information.

   This memo defines a (potentially) vendor-independent interpretation
   of this resource information.

Overview of BOOTP

   While the Reverse Address Resolution (RARP) Protocol [RFC-903] may be
   used to assign an IP address to a local network hardware address, it
   provides only part of the functionality needed. Though this protocol
   can be used in conjunction with other supplemental protocols (the
   Resource Location Protocol [RFC-887], the Domain Name System [RFC-
   883]), a more integrated solution may be desirable.

   Bootstrap Protocol (BOOTP) is a UDP/IP-based protocol that allows a
   booting host to configure itself dynamically, and more significantly,
   without user supervision. It provides a means to assign a host its
   IP address, a file from which to download a boot program from some
   server, that server's address, and (if present) the address of an
   Internet gateway.

One obvious advantage of this procedure is the centralized management
of network addresses, which eliminates the need for per-host unique
configuration files. In an environment with several hundred hosts,
maintaining local configuration information and operating system
versions specific to each host might otherwise become chaotic. By
categorizing hosts into classes and maintaining configuration
information and boot programs for each class, the complexity of this
chore may be reduced in magnitude.

BOOTP Vendor Information Format

   The full description of the BOOTP request/reply packet format may be
   found in [RFC-951]. The rest of this document will concern itself
   with the last field of the packet, a 64 octet area reserved for
   vendor information, to be used in a hitherto unspecified fashion. A
   generalized use of this area for giving information useful to a wide
   class of machines, operating systems, and configurations follows. In
   situations where a single BOOTP server is to be used among
   heterogeneous clients in a single site, a generic class of data may
   be used.

   Vendor Information "Magic Cookie"

      As suggested in [RFC-951], the first four bytes of this field have
      been assigned to the magic cookie, which identifies the mode in
      which the succeeding data is to be interpreted. The value of the
      magic cookie is the 4 octet dotted decimal 99.130.83.99 (or
      hexadecimal number 63.82.53.63) in network byte order.

   Format of Individual Fields

      The vendor information field has been implemented as a free
      format, with extendable tagged sub-fields. These sub-fields are
      length tagged (with exceptions; see below), allowing clients not
      implementing certain types to correctly skip fields they cannot
      interpret. Lengths are exclusive of the tag and length octets;
      all multi-byte quantities are in network byte-order.

      Fixed Length Data

         The fixed length data are comprised of two formats. Those that
         have no data consist of a single tag octet and are implicitly
         of one-octet length, while those that contain data consist of
         one tag octet, one length octet, and length octets of data.

         Pad Field (Tag: 0, Data: None)

            May be used to align subsequent fields to word boundaries

required by the target machine (i.e., 32-bit quantities such
as IP addresses on 32-bit boundaries).

Subnet Mask Field (Tag: 1, Data: 4 subnet mask bytes)

   Specifies the net and local subnet mask as per the standard
   on subnetting [RFC-950]. For convenience, this field must
   precede the GATEWAY field (below), if present.

Time Offset Field (Tag: 2, Data: 4 time offset bytes)

   Specifies the time offset of the local subnet in seconds
   from Coordinated Universal Time (UTC); signed 32-bit
   integer.

End Field (Tag: 255, Data: None)

   Specifies end of usable data in the vendor information area.
   The rest of this field should be filled with PAD zero)
   octets.

Variable Length Data

   The variable length data has a single format; it consists of
   one tag octet, one length octet, and length octets of data.

   Gateway Field (Tag: 3, Data: N address bytes)

      Specifies the IP addresses of N/4 gateways for this subnet.
      If one of many gateways is preferred, that should be first.

   Time Server Field (Tag: 4, Data: N address bytes)

      Specifies the IP addresses of N/4 time servers [RFC-868].

   IEN-116 Name Server Field (Tag: 5, Data: N address bytes)

      Specifies the IP addresses of N/4 name servers [IEN-116].

   Domain Name Server Field (Tag: 6, Data: N address bytes)

      Specifies the IP addresses of N/4 domain name servers RFC-
      883].

   Log Server Field (Tag: 7, Data: N address bytes)

      Specifies the IP addresses of N/4 MIT-LCS UDP log server
      [LOGGING].

Cookie/Quote Server Field (Tag: 8, Data: N address bytes)

Specifies the IP addresses of N/4 Quote of the Day servers
[RFC-865].

LPR Server Field (Tag: 9, Data: N address bytes)

Specifies the IP addresses of N/4 Berkeley 4BSD printer
servers [LPD].

Impress Server Field (Tag: 10, Data: N address bytes)

Specifies the IP addresses of N/4 Impress network image
servers [IMAGEN].

RLP Server Field (Tag: 11, Data: N address bytes)

Specifies the IP addresses of N/4 Resource Location Protocol
(RLP) servers [RFC-887].

Hostname (Tag: 12, Data: N bytes of hostname)

Specifies the name of the client. The name may or may not
domain qualified: this is a site-specific issue.

Reserved Fields (Tag: 128-254, Data: N bytes of undefined
content)

Specifies additional site-specific information, to be
interpreted on an implementation-specific basis. This
should follow all data with the preceding generic tags 0-
127).

Extensions

Additional generic data fields may be registered by contacting:

Joyce K. Reynolds
USC - Information Sciences Institute
4676 Admiralty Way
Marina del Rey, California 90292-6695

or by E-mail as: JKREYNOLDS@ISI.EDU
(nic handle JKR1).

Implementation specific use of undefined generic types (those in the
range 12-127) may conflict with other implementations, and
registration is required.

When selecting information to put into the vendor specific area, care
should be taken to not exceed the 64 byte length restriction.
Nonessential information (such as host name and quote of the day
server) may be excluded, which may later be located with a more
appropriate service protocol, such as RLP or the WKS resource-type of
the domain name system. Indeed, even RLP servers may be discovered
using a broadcast request to locate a local RLP server.

Comparison to Alternative Approaches

   Extending BOOTP to provide more configuration information than the
   minimum required by boot PROMs may not be necessary. Rather than
   having each module in a host (e.g., the time module, the print
   spooler, the domain name resolver) broadcast to the BOOTP server to
   obtain the addresses of required servers, it would be better for each
   of them to multicast directly to the particular server group of
   interest, possibly using "expanding ring" multicasts.

   The multicast approach has the following advantages over the BOOTP
   approach:

    - It eliminates dependency on a third party (the BOOTP server) that
    may be temporarily unavailable or whose database may be incorrect or
    incomplete. Multicasting directly to the desired services will
    locate those servers that are currently available, and only those.

    - It reduces the administrative chore of keeping the (probably
    replicated) BOOTP database up-to-date and consistent. This is
    especially important in an environment with a growing number of
    services and an evolving population of servers.

    - In some cases, it reduces the amount of packet traffic and/or the
    delay required to get the desired information. For example, the
    current time can be obtained by a single multicast to a time server
    group which evokes replies from those time servers that are
    currently up. The BOOTP approach would require a broadcast to the
    BOOTP server, a reply from the BOOTP server, one or more unicasts to
    time servers (perhaps waiting for long timeouts if the initially
    chosen server(s) are down), and finally a reply from a server.

   One apparent advantage of the proposed BOOTP extensions is that they
   provide a uniform way to locate servers. However, the multicast
   approach could also be implemented in a consistent way across
   multiple services. The V System naming protocol is a good example of
   this; character string pathnames are used to name any number of
   resources (i.e., not just files) and a standard subroutine library
   looks after multicasting to locate the resources, caching the
   discovered locations, and detecting stale cache data.

Another apparent advantage of the BOOTP approach is that it allows an
administrator to easily control which hosts use which servers. The
multicast approach favors more distributed control over resource
allocation, where each server decides which hosts it will serve,
using whatever level of authentication is appropriate for the
particular service. For example, time servers usually don't care who
they serve (i.e., administrative control via the BOOTP database is
unnecessary), whereas file servers usually require strong
authentication (i.e., administrative control via the BOOTP database
is insufficient).

The main drawback of the multicast approach, of course, is that IP
multicasting is not widely implemented, and there is a need to locate
existing services which do not understand IP multicasts.

The BOOTP approach may be most efficient in the case that all the
information needed by the client host is returned by a single BOOTP
reply and each program module simply reads the information it needs
from a local table filled in by the BOOTP reply.

Acknowledgments

I would like to thank the following persons for their helpful
comments and insights into this memo: Drew Perkins, of Carnagie
Mellon University, Bill Croft, of Stanford University, and co-author
of BOOTP, and Steve Deering, also of Stanford University, for
contributing the "Comparison to Alternative Approaches" section.

References

    [RFC-951]   Croft, B., and J. Gilmore, "Bootstrap Protocol", Network
                 Information Center, SRI International, Menlo Park,
                 California, September 1985.

    [RFC-903]   Finlayson, R., T. Mann, J. Mogul, and M. Theimer, "A
                 Reverse Address Resolution Protocol", Network Information
                 Center, SRI International, Menlo Park, California, June
                 1984.

    [RFC-887]   Accetta, M., "Resource Location Protocol", Network
                 Information Center, SRI International, Menlo Park,
                 California, December 1983.

    [RFC-883]   Mockapetris, P., "Domain Name - Implementation and
                 Specification", Network Information Center, SRI
                 International, Menlo Park, California, November 1983.

    [RFC-950]   Mogul, J., "Internet Standard Subnetting Procedure",

                    Network Information Center, SRI International, Menlo
                    Park, California, August 1985.

        [RFC-868]   Postel, J., "Time Protocol", Network Information Center,
                    SRI International, Menlo Park, California, May 1983.

        [IEN-116]   Postel, J., "Internet Name Server", Network Information
                    Center, SRI International, Menlo Park, California, August
                    1979.

        [LOGGING]   Clark, D., Logging and Status Protocol", Massachusetts
                    Institute of Technology Laboratory for Computer Science,
                    Cambridge, Massachusetts, 1981.

        [RFC-865]   Postel, J., "Quote of the Day Protocol", Network
                    Information Center, SRI International, Menlo Park,
                    California, May 1983.

        [LPD]   Campbell, R., "4.2BSD Line Printer Spooler Manual", UNIX
                    Programmer's Manual, Vol II, University of California at
                    Berkeley, Computer Science Division, July 1983.

        [IMAGEN]   "Image Server XT Programmer's Guide", Imagen Corporation,
                    Santa Clara, California, August 1986.