

组织：中国互动出版网 (<http://www.china-pub.com/>)

RFC文档中文翻译计划 (<http://www.china-pub.com/compters/emook/aboutemook.htm>)

E-mail: ouyang@china-pub.com

译者：金涛 (piex albertxu@bigfoot.com)

译文发布时间：2001-07-05

版权：本中文翻译文档版权归中国互动出版网所有。可以用于非商业用途自由转载，但必须保留本
文档的翻译及版权信息。

Network Working Group
Request for Comments: 951

Bill Croft (Stanford University)
John Gilmore (Sun Microsystems)
September 1985

引导协议(BOOTP) (RFC951-BOOTSTRAP PROTOCOL (BOOTP))

本备忘录的状态

本RFC文档向ARPA-Internet社区提供一个被提议的协议，需要进一步进行讨论和建议以得到改进。

本文档无发布限制。

目录

1	概述.....	2
2	包格式.....	3
3	鸡和蛋的问题	6
4	ARP在客户端使用	7
5	与RARP对照	7
6	包处理.....	7
6.1	客户端传送.....	7
6.2	客户端重传策略.....	9
6.3	服务器接收BOOTREQUEST (引导请求)	9
6.4	服务器/网关接收BOOTREPLY (引导应答)	11
6.5	客户端接收.....	11
7	通过网关引导	11
8	样例BOOTP服务器数据库	12
9	致谢.....	14
10	参考文献.....	14

1 概述

本RFC描述一种IP/UDP引导协议(BOOTP)，允许一个无盘客户端发现自己的IP地址，服务器主机的地址，和装入一个指定名称的文件到内存并且运行。引导操作有两阶段组成。

本RFC描述第一个阶段：'分配地址和选择引导文件'。

在获得地址和文件名信息后，就进入引导的第二个阶段：文件传送。

文件传送一般使用TFTP协议[9]，因为两个阶段均驻留在客户端的PROM中。

但BOOTP也能够与其它协议如SFTP或FTP一起工作。

我们建议客户端的PROM软件提供一种无须用户交互的完整的引导方式。

这是一种无人值守的上电启动方式。

必须提供一种机制来让用户手工提供地址和文件名信息旁路BOOTP协议直接进入文件传送阶段。

如果提供非可变存储，我们建议在那里保存设置以旁路BOOTP协议直到这些设置导致文件传送阶段失败。

如果缓存的信息失败，引导后退到第一阶段并使用BOOTP。

协议的要点：

- 1.使用了一个单独的包交换（信息）。使用超时机制直到收到应答。

双向使用相同的包字段结构。使用（最大可能长度的）固定长度的字段来简化结构定义和分析。

- 2.一个'opcode'字段包含两个值。客户端广播一个'引导请求(bootrequest)'包。

服务器应答一个'引导应答(bootreply)'包。'bootrequest'包含客户端的硬件地址，如果知道，还包含它的IP地址。

- 3.请求可以包含客户端指定的响应服务器的名称。

这样客户端可以强制从一个指定的主机引导。（如果一个相同的引导文件存在多种版本或服务器在一个远距离的网络/域。）

客户端不必处理名称/域服务，这个功能推到了BOOTP服务器。

- 4.请求可以包含'通用(generic)'引导文件名。例如'unix'或'ethertip'。但服务器发送

引导应答时，它使用对应的引导文件的确切的路径名称来取代这个字段。

服务器查询客户端的地址和请求文件名相关的数据库，以使用客户端自定义的特定引导文件确定这个文件名称。

如果引导请求文件名是空字符串，服务器返回一个带有客户端加载的默认文件的文件名字段。

5.客户端不知道它们的IP地址的情况下，
服务器必须有一个硬件地址和IP地址对应的数据库。
这个客户端IP地址被放在引导应答的（对应）字段中。

6.某些网络拓扑（如斯坦福的网络）可能在一个物理网上没有一个直接可以访问的TFTP服务器

（例如在某些网上的所有的网关和主机都可能是无盘的）。

BOOTP允许客户端通过使用相邻的网关从几跳外的服务器上引导。请看下面'通过网关引导'的章节。

这部分协议不需求客户端部分做特定的动作。
实现是可选的，网关和服务器需要一些额外的代码。

2 包格式

除非另外指出，所有显示的数字都是十进制的。

简化起见，假设BOOTP包不会被分片。

所有数字的字段使用标准网络字节顺序。即，先传送高位比特。

在引导请求的IP头中，客户端如果知道就填自己的IP源地址，否则填0。当服务器地址不知道时，

IP目的地址将是广播地址255.255.255.255。这个地址意味着'在本地网上广播，我不知道我的网络号'[4]。

UDP头包含源和目的端口号。BOOTP协议使用两个保留的端口号，'BOOTP客户端'(68)和'BOOTP服务器'(67)。

客户使用'BOOTP服务器'做为目的端口发送请求；这通常是广播。

服务器使用'BOOTP客户端'做为目的端口发送应答；取决于服务器的核心或驱动设备，这可能是也可能不是广播

(在下面'鸡和蛋的问题'标题的章节中深入解释)。

使用两个保留的端口的原因是当引导应答必须广播到客户端避免'叫醒'并且调度BOOTP服务器进程。

因为服务器和其它主机都不会侦听'BOOTP客户端'端口，

所有进入的广播报文将在核心级别过滤掉。

我们不能简单地允许客户端找一个随机端口号做为UDP源端口字段；因为服务器应答可能是广播，

一个随机选择的端口号可能搞乱其它恰巧在侦听那个端口的主机。

UDP长度字段设置成UDP长度加BOOTP部分的包。

UDP校验和可以由客户端(或服务器)按照需要设置成0，以避免PROM实现中额外的费用。

在下面的'包处理'章节中'[UDP校验和]'短语用来表示校验和可能被验证/计算。

字段	字节数	描述
-----	-----	-----
op	1	packet op code / message type. 包操作码/消息类型 1 = BOOTREQUEST(引导请求), 2 = BOOTREPLY(引导应答)
htype	1	hardware address type, 硬件地址类型 see ARP section in "Assigned Numbers" RFC. 请看 "Assigned Numbers" RFC中的ARP章节 '1' = 10mb ethernet 10M以太网
hlen	1	hardware address length 硬件地址长度 (eg '6' for 10mb ethernet). 例如'6'是10M以太网
hops	1	client sets to zero, 客户端设置成0 optionally used by gateways 在跨越网关引导时网关可选择使用 in cross-gateway booting.
xid	4	transaction ID, a random number,

			used to match this boot request with the responses it generates. 事务ID, 一个随机数, 用来匹配引用请求和应答
	secs	2	filled in by client, seconds elapsed since client started trying to boot. 由客户端填写, 客户端引导开始后的过去的秒数
	--	2	unused未使用
	ciaddr	4	client IP address;客户端IP地址, filled in by client in bootrequest if known.如果客户端知道就在引导请求中填入
	yiaddr	4	'your' (client) IP address;'你的' (客户端) IP地址 filled by server if client doesn't know its own address (ciaddr was 0).如果客户端不知道它的地址 (ciaddr是0), 服务器填入
	siaddr	4	server IP address;服务器IP地址 returned in bootreply by server.由服务器在引导应答返回
	giaddr	4	gateway IP address,网关IP地址 used in optional cross-gateway booting.在跨越网关引导中可以选择使用
	chaddr	16	client hardware address,客户端硬件地址 filled in by client.由客户端填写
	sname	64	optional server host name,可选的服务器主机名 null terminated string. 空结束的字符串
	file	128	boot file name, null terminated string; 引导文件名, 空结束的字符串 'generic' name or null in bootrequest, 在引导请求中使用'通用'名称或空

录路径名称	fully qualified directory-path	是引导应答中使用确切的目录路径名称
	name in bootreply.	
vend 64	optional vendor-specific area,	可选的卖主指定的区域,
	e.g. could be hardware type/serial on request,	例如, 可以是请求硬件类型/序列,
	or 'capability' / remote file system handle	或应答的性能/远端文件系统句柄。
	on reply. This info may be set aside for use	这些信息留给第三方分析引导或核心(程序)使用。
	by a third phase bootstrap or kernel.	

3 鸡和蛋的问题

如果客户端不知道自己IP地址, 服务器怎么发送IP报文到客户端。

无论何时一条引导应答被发送, 发送设备执行下列操作:

1. 如果客户端知道自己的IP地址 ('ciaddr'字段非零), 因为客户端能够回应ARPs [5], 那么IP能够正常发送。
2. 如果客户端还不知道自己的IP地址 (ciaddr是零), 客户端就不能回应引导应答发送程序回的ARPs。这时有两种选择:

- a. 如果发送程序有必需的核心或驱动钩子程序来人工建立ARP地址缓冲条目,

就可以使用'chaddr'和'yiaddr'字段填入一个条目。当然, 这个条目象正常ARP建立的其它条目一样有一个生命时间,

引导应答的发送程序就能够简单地发送引导应答到客户端的IP地址了。UNIX (4.2 BSD)有这种功能。

- b. 如果发送程序缺少这些核心钩子程序, 就只能简单发送引导应答到相应接口的广播地址。

这只是在前面情况外的额外的广播。

4 ARP在客户端使用

客户端PROM必须包含一个ARP的简单实现，例如，地址缓冲能够容纳一个条目。这将允许客户端在知道IP地址和引导文件名后执行第二阶段引导（TFTP）。

任何时候客户端应该准备回应一个自己IP到硬件地址映射的ARP请求（如果知道）以接收TFTP或BOOTP应答。

因为引导应答将包含服务器/网关的硬件源地址（在硬件中封装），客户端可以避免发送一条ARP请求来申请后续的TFTP阶段使用的服务器/网关IP地址。但这应该只是一种特殊情况，因为上面描述的只有第二阶段的引导仍然允许。

5 与RARP对照

提议客户端使用一个早先的协议，反向地址解析协议(RARP) [1]来通过它的硬件地址确定自己的IP地址。

但RARP的劣势是它是一个硬件链路层的协议（不是基于IP/UDP）。

这意味着RARP只能在包含特殊的为访问原始报文修改的核心和驱动的主机上实现。

因为现在存在不同组织维护的许多网络核心，一个不要求修改核心的引导协议是一个确定的优势。

BOOTP除了上述章节描述的有用的特性外，还提供硬件到IP地址的查询功能。

6 包处理

6.1 客户端传送

在第一次建立包前，最好把整个包的缓冲区清零；

这将所有的字段设置成默认状态。任何客户端建立包中的下列字段。

IP目的地址被设置成255.255.255.255（广播地址）或服务器的IP地址（如果知道）。
IP源地址和'ciaddr'设置成客户端IP地址（如果知道），或者0。UDP头使用适当的长度设置；

置；

源端口='BOOTP客户端端口，目标端口='BOOTP服务器端口。

'op'设置成'1'，BOOTREQUEST（引导请求）。'htype'设置成在"Assigned Numbers" RFC ARP章节中分配的硬件地址类型。

'hlen'设置成硬件地址长度，例如，10M以太网是'6'。

'xid'设置成一个'随机'事务ID。'secs'设置成客户端引导开始后过去的秒数。

这个让服务器知道客户端已经试了多长时间了。

当数字变大，某些服务器可能更多注意这个客户端提供不同的服务。

如果客户端缺少一个适当的时钟，它可以使用循环定时器建立一个粗略的估计值。

或者它可以选择简单发送使用一个固定值如100秒的字段。

如果客户端知道IP地址，'ciaddr'（和IP源地址）设置成这个值。

'chaddr'使用客户端硬件地址填写。

如果客户端希望限制从一个特定服务器名引导，就可以在'sname'中放一个空结束的字符串。

使用的名字应该是对应的主机的正当的名字或别名。

客户端在填写'file'文件名字段是有许多选择。

如果设置成空，意味着'我向使用默认的文件来引导我的机器'。一个空文件名也意味着

'我只对找到客户端/服务器/网关的IP地址感兴趣，我不在乎文件名'。

这个字段也可以是一个'通用'名字入'unix'或'gateway'；这意味着

'使用命名的程序配置来引导我的机器'。最后这个字段可以是确切的目录路径名字。

'vend'字段可以由客户端填写卖主的字符串或结构。例如可以填写机器硬件类型或序列号。

但BOOTP服务器的操作应该不依赖与这些存在的信息。

如果使用了'vend', 推荐在'vend'中第一个项目为一个4字节的'魔术字(magic number)'。

这让服务器确定在这个字段中它看到什么类型的信息。

数值可以由通常的'魔术字'过程分配, 你挑一个, 它就成为魔术字。

引导应答使用一个与引导请求不同的魔术字以允许客户端按照应答信息进行特殊的操作。

[UDP校验和]

6.2 客户端重传策略

在一长段时间内没有收到应答, 客户端应该重传请求。

时间间隔必须仔细选择不要引起网络风暴。

可以考虑一个包含100台机器的网络在电源故障后发生的情况。

简单的每四秒重传请求将淹没网络。

一个可能的策略, 你可能考虑指数级的补偿, 象以太网在碰撞时那样。

例如第一个包在0:00, 第二个在:04, 接着:08, 接着:16, :32, :64。

你应该随机化每个时间; 这就象以太网规格那样以一个掩码'与'一个随机数进入第一次补偿。

在每次后续的补偿中, 掩码增长一个比特。

这样在每次补偿中平均延迟加倍。

在'平均'补偿到达60秒后, 就不再增长了, 但仍然随机化。

在每次重传前, 客户端应该修改'secs'字段。[UDP校验和]

6.3 服务器接收BOOTREQUEST (引导请求)

[UDP校验和] 如果UDP目的端口不匹配'BOOTP服务器'端口, 丢弃这个包。

如果服务器名字字段(sname)是空(没有指定特定的服务器), 或者sname是指定的并且匹配我们的名字或别名,

继续包的处理。

如果sname字段是指定的，但不匹配'我们'，那么有多种选择：

- 1.你可以选择简单丢弃这个包。
- 2.如果查询sname的名称显示它在一个网络中，丢弃这个包。
- 3.如果sname在不同的网络中，你可以选择转发这个包到那个地址。

如果这样，检查'giaddr'(网关地址)字段。如果'giaddr'是0，填入我的地址或可以用来到达那个网络的网关的地址。

然后转发这个包。

如果客户端IP地址 (ciaddr) 是0，那么客户端不知道自己的IP地址。

尝试在我们的数据库中查找客户端的硬件地址(chaddr, hlen, htype)。

如果没有匹配，丢弃这个包。否则我们现在对这个客户端有一个IP地址；填入'yiaddr'(你的IP地址)字段。

我们现在检查引导文件名字段(文件)。如果客户端不关注文件名或想要默认引导文件，这个字段是空。

如果这个字段非空，可以将它和客户端的IP地址做为数据库的查询关键字。

如果有默认的文件或通用文件(可能由客户端地址做为索引)或一个匹配的指定的路径名称，

然后在'file'字段中填入选择的引导文件的指定的路径名称。

如果字段是非空并且没有匹配，那么客户端要一个我们没有的文件，丢弃这个包，也许其它BOOTP服务器有这个文件。

卖主指定的数据字段'vend'现在应该检查了。如果提供一种可识别类型的数据，应该进行客户端指定的动作，并且回应要填入应答包中的'vend'数据字段。

例如，一个工作站客户端可能提供一个验证字，并从服务器接收一个访问远端文件的权限，

或一套配置选项传给马上就要导入的操作系统。

我的(服务器)IP地址填入'siaddr'字段。设置'op'字段为BOOTREPLY(引导应答)。

UDP目的端口设置成'BOOTP客户端'。如果客户端地址'ciaddr'非0，把包发送在那里；

否则如果网关地址'giaddr'非0，设置UDP目的端口为'BOOTP服务器'并把包发送到'giaddr'。

否则客户端在我们的一个网络中但它还不知道自己的IP地址，使用在上面'蛋'章节中描述的方法来传送它到客户端。

如果使用'蛋'并且我们在主机上有许多接口，使用'yiaddr'（你的IP地址）字段指出发送到哪个网络（网络/接口）。

[UDP校验和]

6.4 服务器/网关接收BOOTREPLY（引导应答）

[UDP校验和] 如果'yiaddr'（你的[客户端的]IP地址）指向我们的一个网络，使用上述'蛋'方法来将它转发到客户端。

确认将它传送到'BOOTP客户端'UDP目的端口。

6.5 客户端接收

不要忘记为我自己的IP地址（如果我知道）处理ARP请求。[UDP校验和]

客户端应该丢弃以下进入的包：不是定位到引导端口的IP/UDP；不是BOOTREPLY（引导应答）；

不匹配我的IP地址（如果我知道）或我的硬件地址；不匹配我的事务ID。

否则我们就收到一个成功的应答。如果我以前不知道的话，'yiaddr'包含我的IP地址。

'file'是TFTP'读请求'的文件名。服务器地址在'siaddr'中。如果'giaddr'（网关地址）非0，那么包应该先转发到那里，然后到达服务器。

7 通过网关引导

这部分协议是可选的并要求许多网关和服务器配合的额外的代码，但它允许跨越网关引导。这主要在网关是无盘机器时有用。

带盘网关（例如，一个做为网关的UNIX机器）可能运行它们自己的BOOTP/TFTP服务器。

侦听BOOTREQUEST（引导请求）广播的网关可能确定转发还是适当地再广播这些请求。

例如，做为配置表格的一部分，网关可以有一个接收任意BOOTREQUEST（引导请求）广播的其它网络或主机的列表。

即使考虑有一个'hops'字段，简单全部再广播请求仍是一个差的方法，因为广播循环几乎肯定会发生。

转发可以立即开始，或等'secs'（客户端尝试的秒数）字段超过某个阈值。

如果一个网关确定转发请求，它应该查看'giaddr'（网关IP地址）字段。

如果是0，它就在该字段中加入自己的IP地址（在接收的网络中）。

也可以使用'hops'字段来可选控制包可以转发多远。每次转发应该增加跳数。

例如，如果跳数超过'3'，包应该被丢弃。

[UDP校验和]

这里我们推荐在网关中增加这个特殊的转发功能。

但不总是这样子的。

在网上存在一些BOOTP转发代理引导客户端，这些代理可以适当地转发。

这样这些服务可以和网关在一起，也可以不在一起。

当转发代理不和网关在一起时，代理可以通过在接收的引导请求中'giaddr'字段加上接口的广播地址节省一些工作。

这样应答就可以使用普通的网关来转发，而不包含转发代理。

当然劣势是你失去了使用'蛋'非广播方式来发送应答的能力，导致在客户端网上的每个主机的额外的花费。

8 样例BOOTP服务器数据库

做为一个建议，我们提供一个BOOTP服务器查询可以使用的样例文本文件数据库。

数据库有两个节，使用第一列使用一个百分符的行做为定界符。

第一个节包含一个'默认目录'，和从通用名称到目录/路径的映射。

这个节中第一个通用名称是当引用请求包含空'file'字符串是你使用的'默认文件'。

第二节映射硬件地址类型/地址到IP地址。可选的，你可以使用一个特定IP地址的通用名称来忽略默认通用名称。

一个'后缀'项目也是可选的；如果提供，可以访问任意客户端特定的通用名称对应的'路径名称'加上'后缀'。

如果那个文件没有找到，就尝试简单的'路径名称'。

这个'后缀'选项允许轻而易举建立整套用户通用（配置）。

下面显示通用格式；一个或多个空格或TAB来定界字段；后面的空字段被省略；空行和以'#'开始的行忽略。

```
# comment line

homedirectory
genericname1    pathname1
genericname2    pathname2
...

% end of generic names, start of address mappings

hostname1 hardwaretype hardwareaddr1 ipaddr1 genericname suffix
hostname2 hardwaretype hardwareaddr2 ipaddr2 genericname suffix
...
```

这是一个特定的样例。注意'硬件类型'数值同'Assigned Numbers' RFC中ARP章节。
'硬件类型'和'IP地址'数值是十进制；'硬件地址'是十六进制的。

```
# last updated by smith

/usr/boot
vmunix          vmunix
tip             ethertip
watch           /usr/diag/etherwatch
gate            gate.

% end of generic names, start of address mappings

hamilton        1 02.60.8c.06.34.98    36.19.0.5
burr            1 02.60.8c.34.11.78    36.44.0.12
```

101-gateway	1 02.60.8c.23.ab.35	36.44.0.32	gate 101
mjh-gateway	1 02.60.8c.12.32.bc	36.42.0.64	gate mjh
welch-tipa	1 02.60.8c.22.65.32	36.47.0.14	tip
welch-tipb	1 02.60.8c.12.15.c8	36.46.0.12	tip

在上述样例中，如果'mjh-gateway'是一个默认的引导程序，它将得到文件'/usr/boot/gate.mjh'。

9 致谢

Ross Finlayson等早先提出了两个使用讨论RARP[1]的TFTP引导的RFC。

我们感谢Noel Chiappa, Bob Lyon, Jeff Mogul, Mark Lewis, 和David Plummer的前期工作和注解。

10参考文献

1. Ross Finlayson, Timothy Mann, Jeffrey Mogul, Marvin Theimer. A Reverse Address Resolution Protocol. RFC 903, NIC, June, 1984.
2. Ross Finlayson. Bootstrap Loading using TFTP. RFC 906, NIC, June, 1984.
3. Mark Lottor. Simple File Transfer Protocol. RFC 913, NIC, September, 1984.
4. Jeffrey Mogul. Broadcasting Internet Packets. RFC 919, NIC, October, 1984.
5. David Plummer. An Ethernet Address Resolution Protocol. RFC 826, NIC, September, 1982.
6. Jon Postel. File Transfer Protocol. RFC 765, NIC, June, 1980.
7. Jon Postel. User Datagram Protocol. RFC 768, NIC, August, 1980.

8. Jon Postel. Internet Protocol. RFC 791, NIC, September, 1981.
9. K. R. Sollins, Noel Chiappa. The TFTP Protocol. RFC 783, NIC, June, 1981.