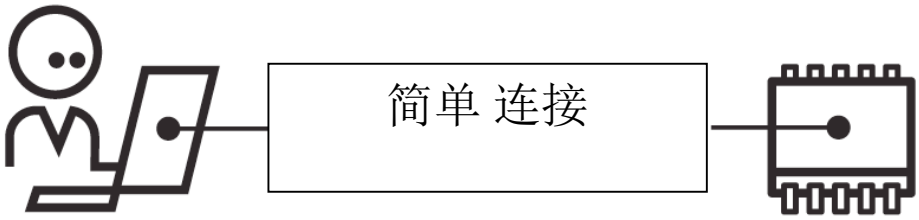
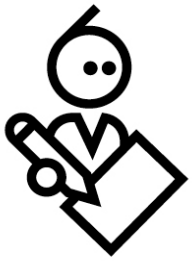


Nabto平台简介

NABTO/001/TEN/029



目录

1 摘要

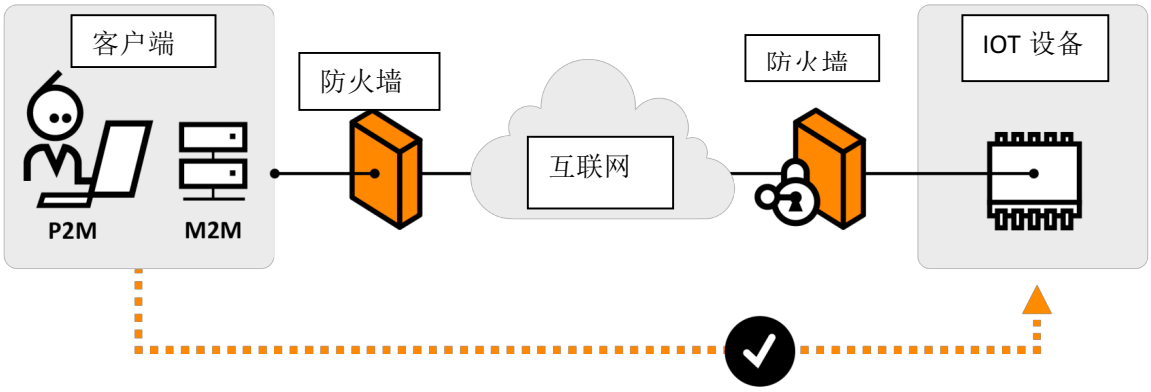
本文件总结了Nabto平台的相关信息，包括集成选择，性能指数以及目标平台要求。

2 参考文献

TEN017	NABTO/001/TEN/017： uNabto SDK – 编写源代码
TEN023	NABTO/001/TEN/023： uNabto SDK – 编写Nabto设备应用
TEN024	NABTO/001/TEN/024： uNabto SDK –编写Nabto HTML客户端应用
TEN025	NABTO/001/TEN/025： uNabto SDK – 编写Nabto API客户端应用

3 Nabto 是什么？

Nabto提供一个全通信基础设施，即Nabto通信平台，让客户端以及即使资源非常有限的设备之间实现直接加密的通信。该平台支持点对点通过NAT遍历实现直接连接。如果两点的防火墙均不允许该连接，可使用透明中继（中转）建立连接。



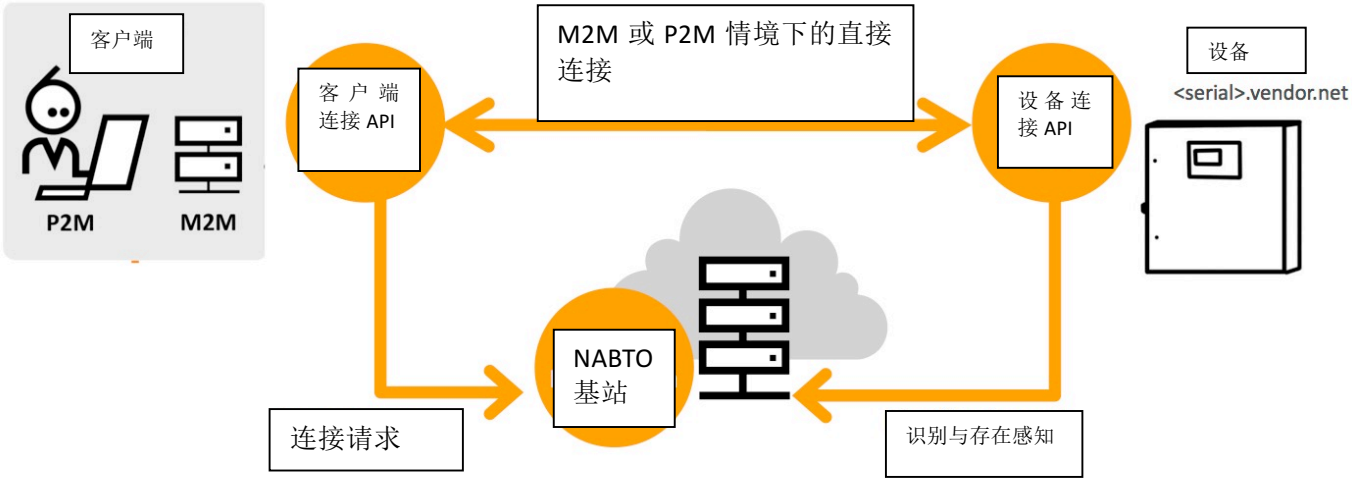
- 供应商结合Nabto高度优化内置软件：10kB 闪存，需要2kB RAM（参见“设备要求”章节）[1]
- 各个设备在DNS中均有一个唯一ID。例如：<serial>.<vendordomain>.net

- 无论在哪里均可以无缝安全地“呼叫”设备，要求发送数据，发布指令或启动数据流

- 线上线下两种环境：通过云实现的Skype™ 模式或Bonjour™ 本地通信模式
- 云只储存运行时数据：确保高度隐私性和极端扩展性
- 互动性网络体验：：Nabto完全融入浏览器，使用方便，采用率高，升级简单
- 客户端和设备之间安全透明的数据隧道

[1] 如果手动编辑执行程序并越过SDK提供的抽象层，则可在低于1kB的闪存条件下执行uNabto网络协议

4 Nabto 平台的组成部分



Nabto平台由三部分构成:

- Nabto**客户端**： Nabto提供的二进制，客户的HTML或本地应用使用
- Nabto**设备**： uNabto SDK-Nabto提供的开源框架，与客户的设备应用结合
- Nabto**基站**： Nabto提供的服务（Nabto或自托管），调节Nabto客户端和设备之间的连接。同时向Nabto HTML客户端提供用户界面。

Nabto发起与支持Nabto的设备的直接加密连接–Nabto基站调节该直接连接：设备的唯一名称，比如<serial>.vendor.net映射到Nabto基站的IP地址–Nabto基站是设备联网时注册以及客户端寻找可用设备的地方。连接建立后，基站便“置身事外”-没有数据储存在基站内，它只知道当前可用的支持Nabto的设备。

客户端还可发现设备是否位于同一局域网，是否可以无需通过基站进行直接通信-这对于辅助程序情境或脱机情景使用时很有效。

将Nabto与客户设备结合请参见[TEN023]。

客户的客户端应用可通过不同的方法使用Nabto客户端：客户应用可以作为HTML应用，使用Nabto客户端在网络应用中检索JSON数据-该种情境下，Nabto客户端通常为一个网络浏览器插件或移动应用软件，托管客户应用。后者为从基站分配到客户端，表示为一个HTML设备驱动程序包。该应用的编写请参见[TEN024].

客户的客户端应用也可以是本地（非HTML）应用，与Nabto客户端API库连接。本地客户端应用可

使用与HTML应用相同的请求/回复机制调用设备。而且，本地客户端可建立与设备的数据流连接-这是向旧版客户端和设备应用添加无缝安全远程访问功能的常用做法。本地客户端应用请参见[TEN025].

5 支持的客户端

5.1 HTML 客户端应用

插件或应用执行客户HTML应用。插件和应用均可与客户主题（应用商店的标识，文本，名称，具体视情况而定）结合来创造新名称。

Microsoft Windows（32/64比特，Win XP SP3及更新版）	IE浏览器插件（所有版本） 火狐扩展（3个最新发布的）
Mac OS X	火狐扩展（3个最新发布的）
Linux （32/64比特）	火狐扩展（3个最新发布的） 仅支持官方摩斯拉浏览器（非特定发行版）
Android 3.x及更新版	运行Nabto HTML应用的专用应用
iOS 4.x及更新版	运行Nabto HTML应用的专用应用
Windows Phone 8.0及更新版	预计2015年1季度将提供Windows店铺应用。那时，Windows 8 X86的平板将可以下载安装应用。
代管客户端	任何浏览器，无需插件-但要求存在网络，且代管客户端组件安装在基站

编写HTML客户端应用信息请参见[TEN024]。

请注意目前并没有支持谷歌chrome浏览器或苹果safari浏览器的HTML客户端插件。在其出现前，可以使用上述托管客户端解决方案-而限制就是只能在互联网连接情况下使用，且不支持本BONJOUR模式探索。仍然可以为所有使用Nabto客户端API（在下文描述）的浏览器建立自定义浏览器组件-如建立自定义视频流ActiveX或NPAPI组件。

5.2 本地客户端应用

Nabto客户端API可以作为一个基础C库，并可使用平台上的所有功能。此外，提供一个面向对象的.NET库，将低层次的API打包进.NET平台使用的典型抽象层-比如，它可以替代从专利客户端/服务器执行升级到使用Nabto的应用中的传统网络流对象。

Nabto数据流功能（比如视频流）只能通过Nabto客户端API实现（HTML客户端无法使用）。

Microsoft Windows （32/64比特）	C库，.NET4.0抽象层
Mac OS X	C库，.NET4.0抽象层 （需要单声道）
Linux （32/64比特）	C库，.NET4.0抽象层 （需要单声道）
Android 3.x及更新版	C库，带JNI包装程序
iOS 4.x及更新版	C库
Windows Phone 8.0及更新版	因平台限制，没有库支持

6 支持的设备

嵌入式设备的Nabto SDK （uNabto[2] SDK） 作为开放源对设备供应商开放。
基本上Nabto设备通常由以下组件构成：

- uNabto**框架**：将所有复杂问题比如安全性和NAT遍历抽象化。完全由Nabto提供，供应商可进行设置。
- uNabto**平台转接器**： uNabto框架和相关设备平台之间的连接器，让uNabto框架能够收发UDP包等。Nabto提供数个转接器作为开源SDK的一部分（见以下），供应商可为不支持的平台使用转接器。
- uNabto框架和供应商后端应用之间的连接器（比如根据客户端请求调用后端）。由供应商执行。

Nabto提供一组平台转接器，可以直接使用或作为新的供应商特定转接器的基础：

RAKWireless	RAK415 和 LX520
Microsoft	WIN32 （x86和x64）, Windows CE
Linux	任何Linux和uClinux 变体，只需要一个交叉编译工具链
FreeRTOS	通过FreeRTOS+完全统合
MicroChip	PIC18和PIC32
Freescale	ColdFire
Renesas	RL78和RX600
Atmel	AVR gcc
Quectel	M10
RTX	RTX4100和RTX4140
Gainspan （芯片上）	GS1100
Gainspan （按要求）	GS1100和GS1500
Arduino	

[2] 称为*微-Nabto*

Mbed	NXP LPC1768 （Cortex M3）

uNabto设备应用的组件相关信息请参见[TEN023] 。

uNabto设备应用的资源要求很大程度上视目标架构和所需特征而定。平台是以模块为基础的，因此可以从应用中省略一些特征以节约内存/闪存。uNabto源代码设置的相关信息请参见[TEN017] 。

7 安全性

Nabto平台使用X509/PKI进行客户端认证以及在客户端和设备间启动安全通信通道。设备使用共享密钥认证（HMAC-SHA256/AES-128），建立一个安全的通信通道返回客户端。基站发挥调节作用，将经认证的客户端的识别信息发送至设备，并在客户端和设备之间交换会话密钥以实现数据机密性。

完全PKI 安全性（vs.共享密钥设备安全性）将作为后期平台的一个特征（按客户要求进行优先化）。

在三个层面上进行访问控制：

1. 基站上的粗粒度访问控制：是否允许连接的客户端接入请求域的设备？
2. 设备上的连接层面访问控制：设备接受来自基站的连接客户端的加密信息，然后将其与设备上的访问控制名单进行对比。
3. 设备上的功能层面访问控制：为了启用设备各个功能，客户端将提供其加密信息。而设备将把该信息与设备上储存的授权矩阵进行对比。

访问控制由uNabto平台上的基础机制（访问识别和连接信息）以及作为SDK一部分提供的应用模块（保存访问控制和特权名单）支持。基础机制和供应的模块的详细信息请参见[TEN023]。

8 网络

8.1 点对点支持

Nabto平台确保直接的点对点连接可以在任何理论可行的网络设置中建立。如果两点的防火墙均不支持UDP打洞，将使用一个透明中继（中转）建立连接。这对客户端应用完全透明，尽管本地客户端应用可以查询实际连接类型（比如断开长期中继，如果连续播送 HD 视频）。

下表说明了可能的组合：如果一个字段含有"ok"，则可建立点对点连接。比如，可在端口限制的NAT后面的点和地址限制的NAT后面的点之间建立点对点连接。实际操作中无法在对称型NAT后的一个点和另一个对称型NAT后的设备之间建立连接（尽管理论上可行）。

	全锥形	地址限制	端口限制	对称型	开放型
全锥形	**ok**	**ok**	**ok**	**ok**	**ok**
地址限制	**ok**	**ok**	**ok**	**ok**	**ok**
端口限制	**ok**	**ok**	**ok**	**ok**	**ok**
对称型	**ok**	**ok**	**ok**	**fail**	**ok**
开放型	**ok**	**ok**	**ok**	**ok**	**ok**

以上不同类型间的互联网防火墙并不是均匀分布的。Nabto的经验发现防火墙的数量在客户以及工业/公司之间相差很大（通常是因价格低廉而更加简单），而且在不同国家以及不同终端用户客户端之间（移动vs. 固定ADSL/电缆）相差也很大。

以上图表是来自消费者和工业应用的实际数据。所收集的数据主要来自美国。

系列中防火墙类型的整体分布：

观察	比例
	总比例
全锥形	15,36
地址限制	10,47
端口限制	51,29
对称型	21,34
开放型	1,54

相当于以下（P2P）成功矩阵：

概率 （成功 – P2P）						
	全锥形	地址限制	端口限制	对称型	开放型	**总计**
全锥形	2,4	1,6	7,9	3,3	0,2	**15,4**
地址限制	1,6	1,1	5,4	2,2	0,2	**10,5**
端口限制	7,9	5,4	2	10,9	0,8	**51,3**
对称型	3,3	2,2	10,9	0,0	0,3	**16,8**
开放型	0,2	0,2	0,8	0,3	0,0	**1,5**
总计	**15,4**	**10,5**	**51,3**	**16,8**	**1,5**	**95,4**

以及以下失败（中继）矩阵：

概率 （失败 – 中继）						
	全锥形	地址限制	端口限制	对称型	开放	**总计**
全锥形	0,0	0,0	0,0	0,0	0,0	**0,0**
地址限制	0,0	0,0	0,0	0,0	0,0	**0,0**
端口限制	0,0	0,0	0,0	0,0	0,0	**0,0**
对称型	0,0	0,0	0,0	4,6	0,0	**4,6**
开放型	0,0	0,0	0,0	0,0	0,0	**0,0**
总计	15,4	**0,0**	**0,0**	**4,6**	**0,0**	**4,6**

完全消费者设置中P2P连接成功率会更高。

8.2 IPv6 支持

Nabto平台目前不能本地支持IPv6；所有网络互动目前都基于IPv4 。计划2015年第二季度将支持IPv6。

8.3 uNabto 设备网络环境

uNabto设备需要*出站*网络接入Nabto基站的两个UDP端口（一个给定的设备名称被当作主机-比如demo.nabto.net被当作195.249.159.159上的演示基站）。默认配置如下：

- 基站的控制器服务： UDP 端口5566
- 基站的uDirectory服务（GSP）： UDP端口5562
- 基站的 TCP 网关（如果设备需要TCP 中继）： TCP端口5568

为能够建立点对点连接，设备需能够通过其防火墙将数据包发送给任何UDP主机和端口。

8.4 客户端网络环境

Nabto客户端需要出站网络接入Nabto基站上以下子集的端口（默认端口号）：

- 基站的 STUN服务： UDP 端口3478
- 基站的控制器服务： UDP端口5566
- 基站的TCP 网关： TCP端口 5568
- 基站的HTTPS服务： TCP 端口 443
- 基站的HTTP 服务： TCP端口80

理想情况下，所有这些设备都需要通过防火墙进行出站访问，但是如果只需要部分功能，Nabto各点也将在更多限制的配置下工作，如下表所示。

此外，为能够建立点对点连接，设备需能够通过其防火墙将数据包发送给任何UDP主机和端口。

客户端端口打开进行出站访问	STUN UDP	控制器 UDP	网关 TCP	HTTPS TCP	HTTP TCP	全开 UDP
P2P连接	是	是	否	否	否	是
TCP中继转发连接	否	是	是	否	否	否
HTTP中继转发连接（仅限客户端）	否	否	否	否	是	否
HTML设备驱动的初始安装	否	否	否	是	是	否

9 服务站

如前所述，Nabto 服务站在Nabto整体解决方案的一个中央服务中心。所有设备在Nabto服务站进行注册，确保客户端发起与某一设备端到端连接请求时能够顺利找到该设备。Nabto服务站起到一个中间调节的作用，客户端发来的与设备端到端连接请求经过Nabto服务站处理之后与设备建立直接的P2P 连接，如果P2P连接不成功Nabto服务站还能起到一个数据转发的功能，Nabto服务站具有信息转发功能。

Nabto服务站可以当作纯粹的 SaaS 来使用，产品生产商无需担心部署，分类以及划分物理区域等问题，我们的Global Nabto Cloud 服务站可以解决所有问题。用户还可以购买整个服务站，完全由自己管理。

9.1 全球 Nabto Cloud

在 Global Nabto Cloud SaaS 服务站部署上，Nabto 提供一组服务器，这些服务器遍布世界，用以最大化可利用率和做好的使用效果。 产品生产商无需担心用户将要在世界那个角落使用产品，因为每一个设备都会自动寻找距离自己最近的数据中心服务器并且在该服务器注册，用户 将会自动的被引领导正确的服务站。如果用户的产品位置改变了，该产品会在新的地址在距离自己最近的另外一个数据中心的服务站自动注册。

产品生产商无需担心负荷平衡的问题，因为数据中心服务站会自动调节容量来保持P2P连接和数据转发性能，无需担心工作量的问题。

Global Nabto Cloud 能与所有的Nabto 协议版本兼容，所以不管现有的产品还是以后的新产品都可以使用并从中受益。目前的基础设施包括数据中心在EU, US 和中国大陆。如果在其他区域有特殊要求，Nabto 可以在该区域增加服务器。

如果用户追求简化P2P连接的整体解决方案，降低成本，并针对用户可使用率和实际使用性能来最优化最终用户使用体验的话，我们推荐这种方式来与产品生厂商合作。然而，一些客户有完全私有服务的需求，也就是说不想把自己的产品和其他用户的产品在一起由Nabto统一管理，Nabto还可以针对这一部分客户的要求提供单独的服务站，详情请看下方文字 。

9.2 私有基站

现在，许多用户如果不想使用全球Nabto Cloud服务的话，也可以自己单独管理自己的服务站，或者直接从Nabto购买私有服务站管理服务让Nabto来代为管理。这个模式在以下文字有详细解释。所有私有云的构建和以上提到的全球Nabto Cloud 迷失完全一样，唯一不同的是所有服务器和服

务站完全私有。

9.2.1 容量

Nabto基站容量根据选定的Nabto许可和主机可获得的资源（内存，CPU，网络带宽）而定。各个联网设备均要求在基站存在大约10 kB 的内存。我们的参照主机平台为一台Amazon EC2 小型实例（单核），经测试可以处理10,000 台设备。Amazon EC2 c3.x大型实例（4核）经测试可以处理100,000 台设备。

在基站注册的各个设备默认每10秒发送一条实时信息：向基站发出并从基站收到25比特。这相当

于每个月每台设备12 MB。以EU Amazon EC2为例，这意味着100,000台设备每个月的闲置流量费用为70美元（2014年4月价格）。而实际使用的成本将高于该数值（比如连接请求，中继流量， HTML设备驱动更新）。

9.2.2 运行/ 部署

Nabto基站是一组在公共IP地址上运行的服务，听取几个UDP和TCP端口命令。用于生产目的的基站目前可在Linux类型系统获得支持。基站软件同样也可在Windows和Mac OS X上运行（在任何其他Unix变体上可能需要轻微的改编）。但是目前没有结合管理/监控服务-可按客户要求添加。

Nabto基站可以托管在客户自己的服务器环境， Nabto的主机设备或云上-其在各类VPS提供商，包括Amazon EC2，均可运行良好。

可以通过DNS将负载分布到各个基站上-通过改变DNS映射，可以动态变更设备注册的基站。

基站没有保持任何持续状态，因此可以通过热备份进行失效备援：备用实例联网后，所有设备在新实例上重新注册，然后在重建失效备援前保持状态。

9.2.3 服务站的管理

Nabto 可以帮助用户管理服务站。 管理工作需要使用Nabto的数据中心或使用亚马逊的AWS平台。

又两种服务可供用户选择。一种是标准的管理服务不包含SAL（Service Level Agreement），不使用自动失效备援机制。另外一种是Nabto的服务站高级托管服务。

高级托管服务带有SLA可以享受以下保证:

- 每个月可以保证99.95%的正常运行
- 每月正常运行低于以上百分比的话，每多于1分钟的非正常运行时间，Nabto 按照每分钟抵扣1% 每月管理费用予以赔偿。最多可以抵扣月管理费的25% 。

高级托管服务SLA适用条件:

- ”非正常运行” 指的是服务站所处状态服务站通过所注册的设备名称不能使用户接入目标设备。
- 此功能需要客户允许Nabto在亚马逊的AWS平台上运行服务站。
- 控制服务窗口不影响SLA服务
- 如果在亚马逊上服务器在同一时间不同区域同时不能使用的情况下SLA不适用。DoS 攻击服务器导致的状况也不在SLA范围之内。

高级托管服务的价格比标准托管服务的价格高25%。详情请看http://nabto.com for more info on pricing or contact sales@nabto.com.

10 数据流性能与限制

如 [TEN025]所述，使用Nabto TCP隧道或原始Nabto流时，其吞吐量由以下参数决定：

- Nabto流MTU大小 - 截至本说明书编写时间，默认为1311 比特 （**mtu**）
- 点之间的往返延迟- 直接连接或通过基站进行中继连接 （**rtt**）
- Nabto 流窗口大小-（**win_size**）

理想的且没有数据包丢失，复制或重新排序的情况下的理论吞吐量：

吞吐量 = mtu_size * win_size / rtt

比如，如果两点之间的往返时间为200ms,默认窗口大小为100 yields，则吞吐量为约 5 Mbps。在中继情况下，rtt通常翻倍，这意味着预计吞吐量降低至2.5 Mbps.

Nabto流的当前执行存在以下限制：

- 优化设备到客户端的吞吐量性能是不对称的（即，在标准视频流情况下性能最佳，而在比如推动固件升级时性能将受限制）。
- 窗口大小是固定的（编译时设置），开发过程中须做最坏准备：实际网络状况下可能无法收到反馈。
- 如果CPU功率有限的平台上的窗口过大，则隧道可能占用过多的CPU。

须进行仔细分析和测试来平衡目标平台上的窗口大小，吞吐量要求， CPU以及内存占用情况。

Nabto 平台计划在 2014 年通过高端平台（比如 Linux 支持设备）克服这些限制（不对称性和固定的窗口大小）。