

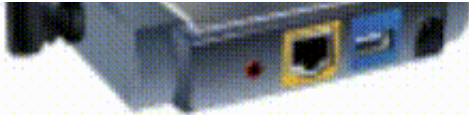
OTA概述

大家好，我是一个软件升级包。这几天呢，我将会进行一次神奇的网络之旅，从开发者的电脑中，一直跑到终端嵌入式设备中。

大家都把我的这个旅游过程叫做 OTA，也就是在线升级。

那么啥叫 OTA 呢？全称是：Over the Air Technology，其实就是通过网络来把一个新的软件包从服务器上下载下来，更新到设备上。





下载压缩包

首先有一个问题：为什么叫**软件**升级包，而不叫**固件**升级包呢？

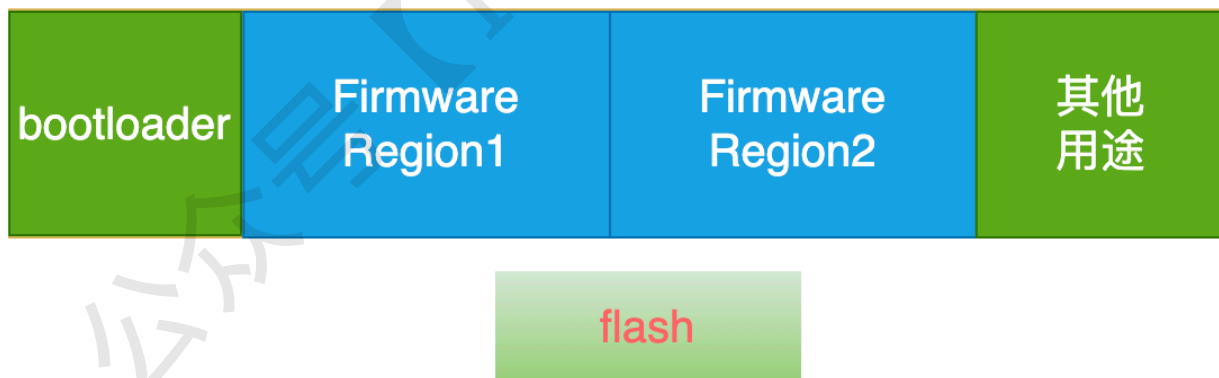
其实在本质上，固件也是属于软件，大家都是用代码写出来的嘛！

虽然这两个说法很近似，但是有一部分小伙伴还是在**狭义**上对它们进行了一些区分。

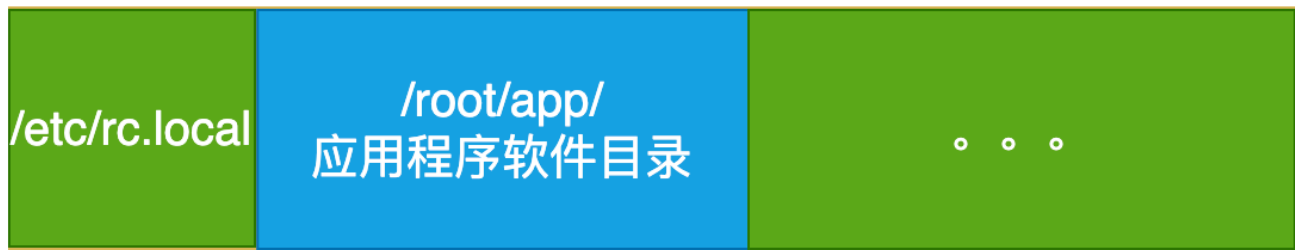
既然如此，我们也就暂且把它俩进行一下区别：

1. 固件：是指一些没有文件系统的嵌入式设备中，把 Flash 分成不同的功能分区。可执行程序需要放在某个固定的起始位置，才能被 bootloader 进行启动。
2. 软件：是指具有文件系统的嵌入式设备，可执行程序直接放在文件系统中。当设备启动之后，操作系统会启动文件系统中的可执行程序。

没有文件系统的嵌入式设备：



带有文件系统的设备：



文件系统

我知道以上这样的区分方式不是很严谨，但是谁又说得清楚严谨的定义是什么呢？

暂且先这么来区分，只要不影响对文章的理解就可以了！

一个嵌入式设备在进行软件升级的时候，从宏观的角度看，可以分为2个阶段：

1. 下载升级包；
2. 解压升级包，写入 flash 或文件系统；

今天呢，主要以第1阶段为主，带你看一下我是如何从开发者的电脑里，一步一步的被嵌入式设备下载到本地的。

下面是一个完整的过程，让您先睹为快！



上传升级包

为了便于描述，我们来假设一个场景：运行在设备中的软件一共有 3 个文件：

1. main 文件：可执行程序；
2. config.ini：配置文件；
3. mylib.so：一个动态库文件，里面包含一个算法，被 main 文件调用；

目前呢，设备中运行的版本是 V1.0，现在开发人员对 mylib.so 库中的算法进行了优化，升级为 V2.0 版本，现在需要把这个新版本升级到嵌入式设备中。

公众号【IOT物联网小镇】

首先第一步需要做的事情，咱们用脚后跟都能想得到，那就是把 V2.0 版本的程序软件上传到文件服务器中。

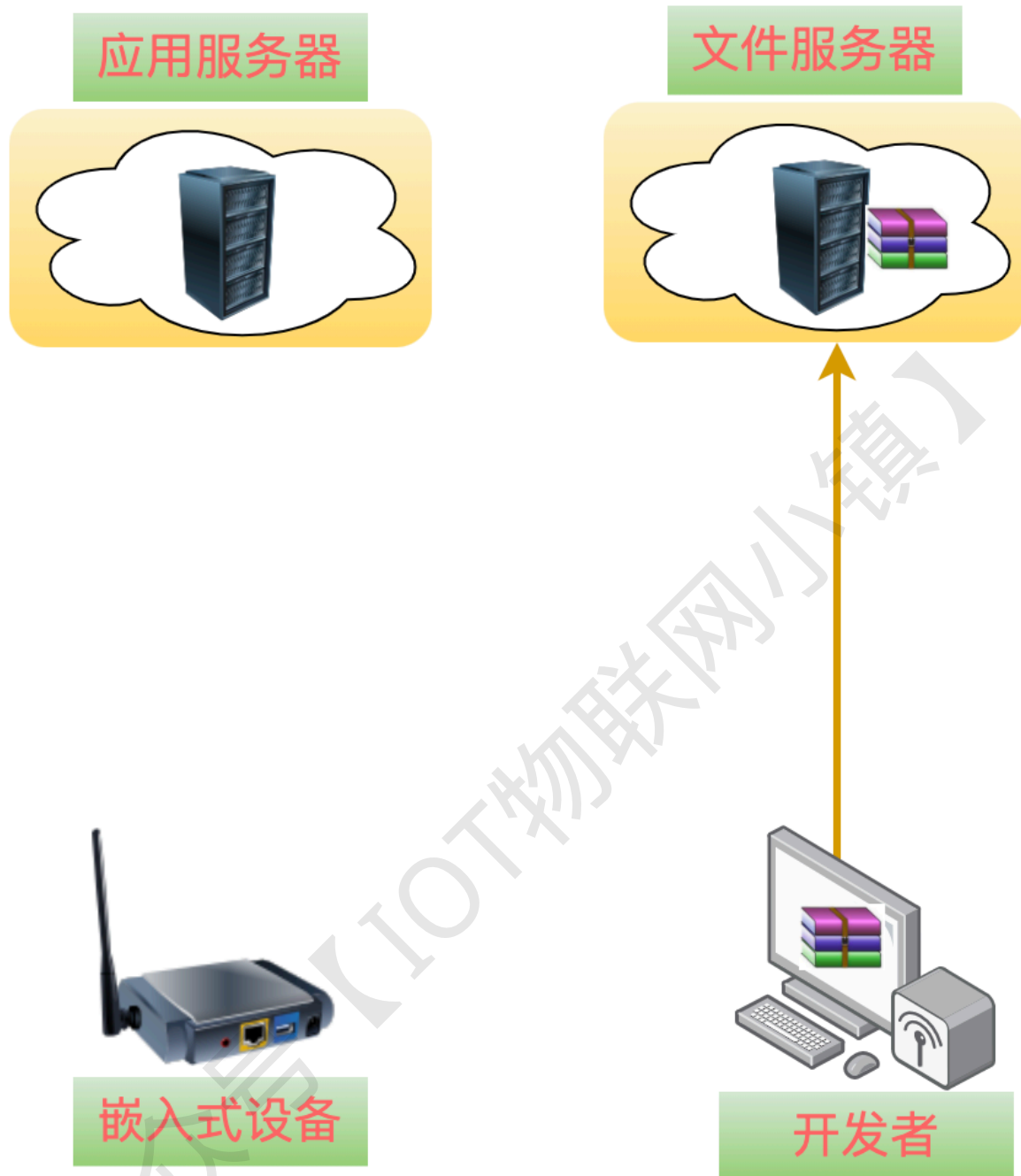
有一点提醒一下：很多云平台都会把应用服务器和文件服务器进行区分。当然，如果仅仅是测试的话，它俩可以在同一台物理服务器上共存。

比如：亚马逊的 AWS 平台，就是把升级包上传到 S3 服务器中。

现在要对 V2.0 版本的程序进行打包了，在这里，除了 main、config.ini、mylib.so 这 3 个文件之外，我们还把另一个脚本文件 upgrade.sh 也放进打包文件中。

这个文件的作用暂且不说，到后面会为您揭晓答案。

Bingo - V2.0 版本的升级包诞生了：app_v2_0.tgz，上传到文件服务器上之后，地址为：http://fileservce/app_v2_0.tgz。



上传升级包描述文件

现在，V2.0 版本的升级包已经上传到文件服务器中了，是否现在就可以命令嵌入设备去下载、升级了呢？

我们知道，在一个物联网系统中，一般都是存在着很多个终端设备的。

这些设备可能处于正在运行状态、也可能处于断电状态，而且咱们也不能假设所有的设备都在同一个时间点进行升级。

公众号【IOT物联网小镇】

再而且，一个设备进行升级之后，就变成了最新的 V2.0 版本，那么这个设备就应该有能力知道服务器上的最新版本是 V2.0 版本，这样它就不需要升级了。

因此，还需要一个新的文件来描述文件服务器中的 V2.0 版本的升级包，就叫它：升级包描述文件 app_desc.json，它的内容是 json 格式的字符串：

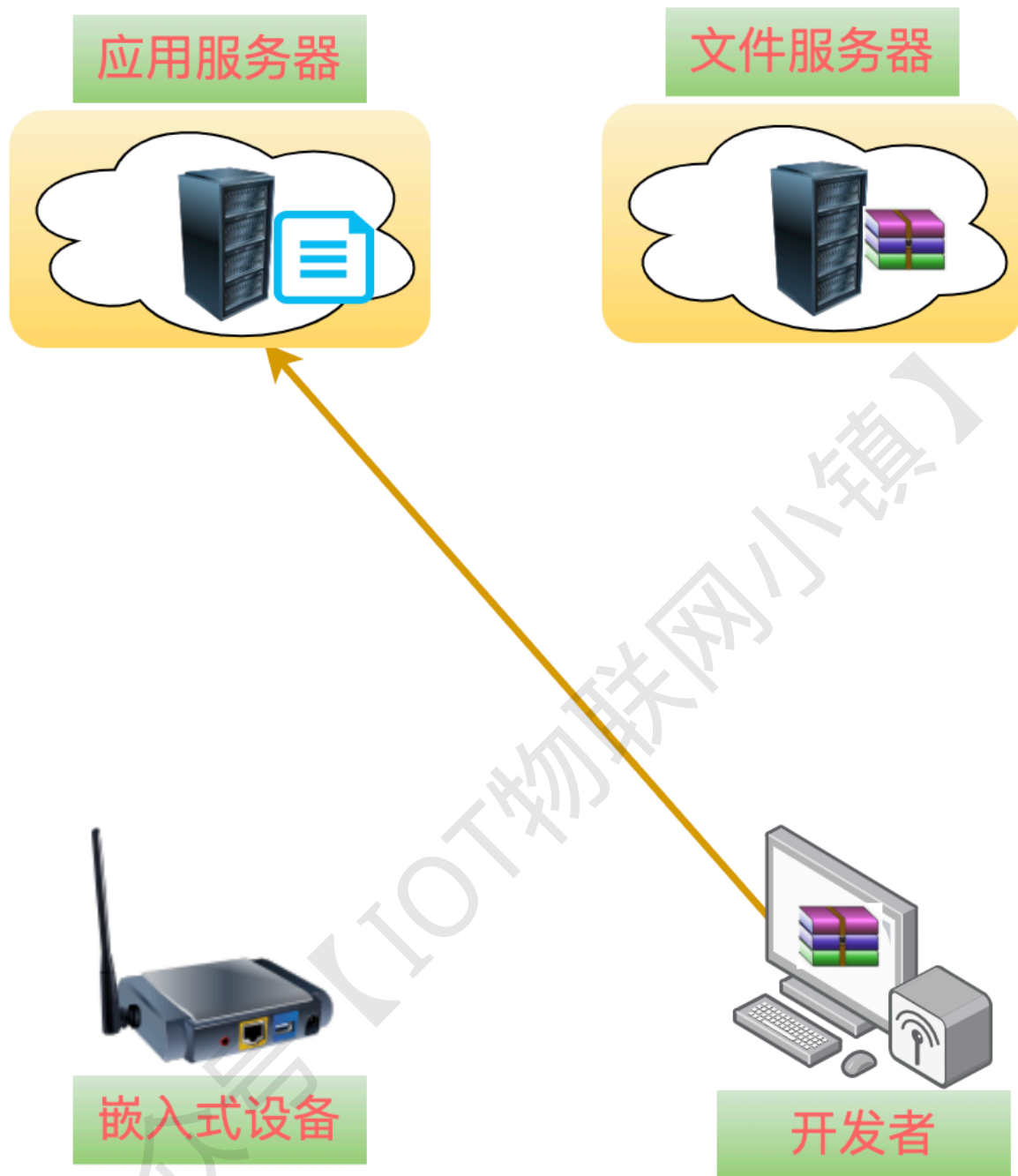
```
{  
  "version": "v2.0",  
  "url": "http://fileserve/app_v2_0.tgz",  
  "md5": "xxxxxxxxxxxx"  
}
```

version 字段描述了文件服务器升级包的版本，这样的话，设备就可以知道到服务器中的最新版本。

url 字段描述了升级包的下载地址，设备如果发现自己的版本低于 version 字段中的版本，就可以从这个地址下载新的升级包。

md5 字段描述了服务器中最新升级包的指纹信息，当设备把服务器上的升级包下载之后，需要计算一下升级包的 MD5 值，然后与这里的 md5 字段进行比较，如果相同的话，说明下载的升级包没有问题，没有被恶意的家伙掉包。

了解了升级包描述文件 app_desc.json 的作用之后，这个文件就被上传到应用服务器中了。



下载升级包描述文件

此时，作为升级包的我，已经静静的躺在文件服务器中了，我的兄弟升级包描述文件 `app_desc.json` 呢，也在应用服务器中准备就绪了，现在就等着嵌入式设备开始升级。

万事俱备，只欠东风了！应该说只欠一个触发嵌入式设备进行升级的动作了！

那么，应该在什么时候？由谁？来告诉设备：你正在运行的软件太旧了，服务器上现在有最新的版本，你去升级一下吧！

公众号【IOT物联网小镇】

这个问题的答案就是：八仙过海，各显神通了！

比如：

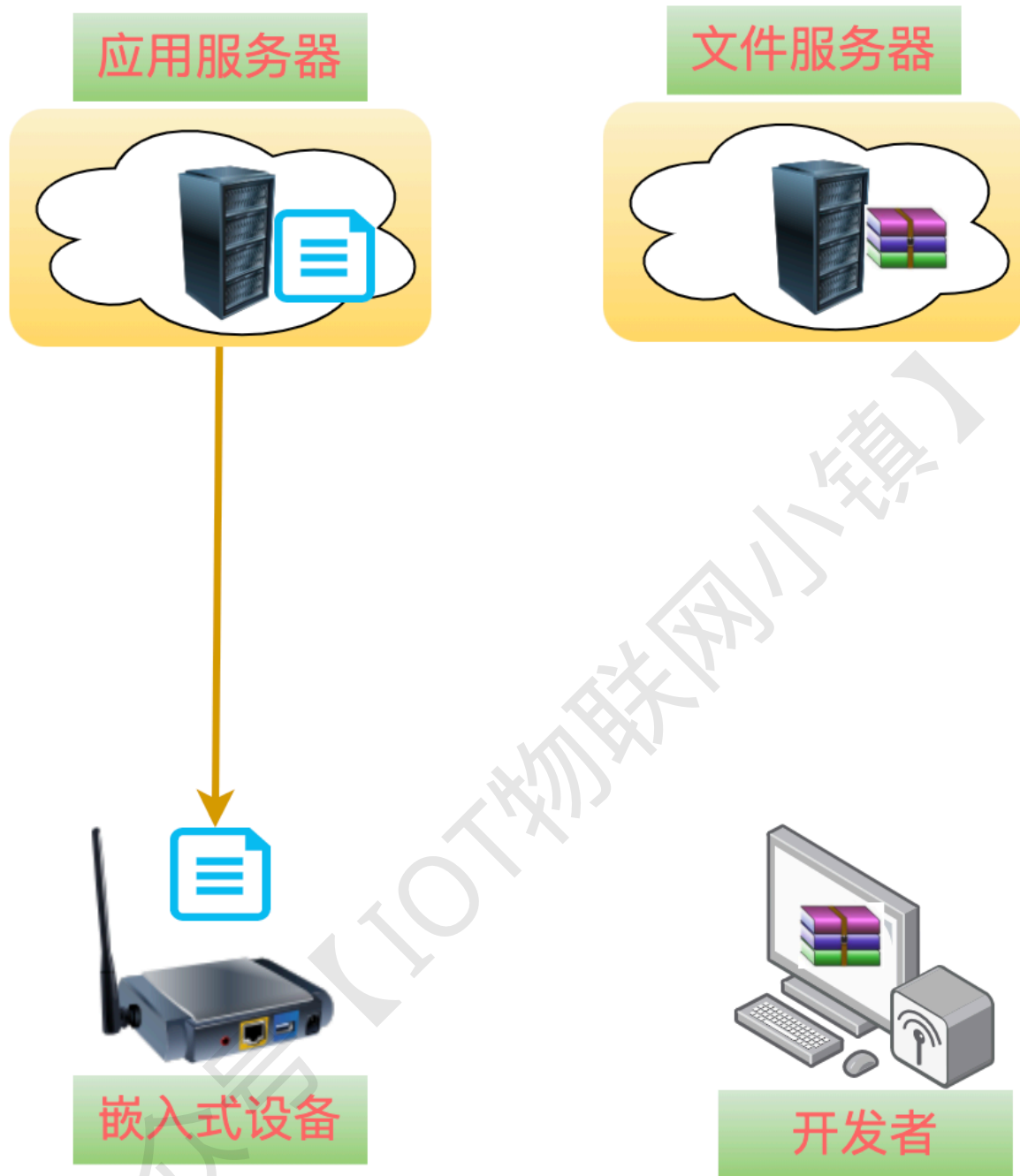
1. 亚马逊的 AWS 平台，是通过在云平台中部署一个 job，来通知每一个需要升级的设备；
2. 也可以通过一个手机 APP，向某一个嵌入式设备主动发起一个指令：嘿，老兄，请升级一下你的软件；

AWS 平台中，在把**升级包**和**描述文件**上传到服务器上之后，还需要添加一个 **job**，这个 **job** 中包含如下信息：

1. 升级包描述文件的地址；
2. 需要升级的终端设备。

当 **job** 添加好之后，这个 **job** 就会被**推送**给选择的所有终端设备(通过 **MQTT** 的订阅机制)，这样一来，就进入下面的下载流程了。

当终端设备收到升级命令之后，第一步就是下载**升级包描述信息**。



下载之后，解析这个 json 格式的文本内容，提取出 version 信息之后，与当前正在运行的软件版本进行比较。

如果服务器中的版本比较新，那么就继续提取 url 字段中的升级包下载地址，然后开始从文件服务器中[下载](#)新的升级包。

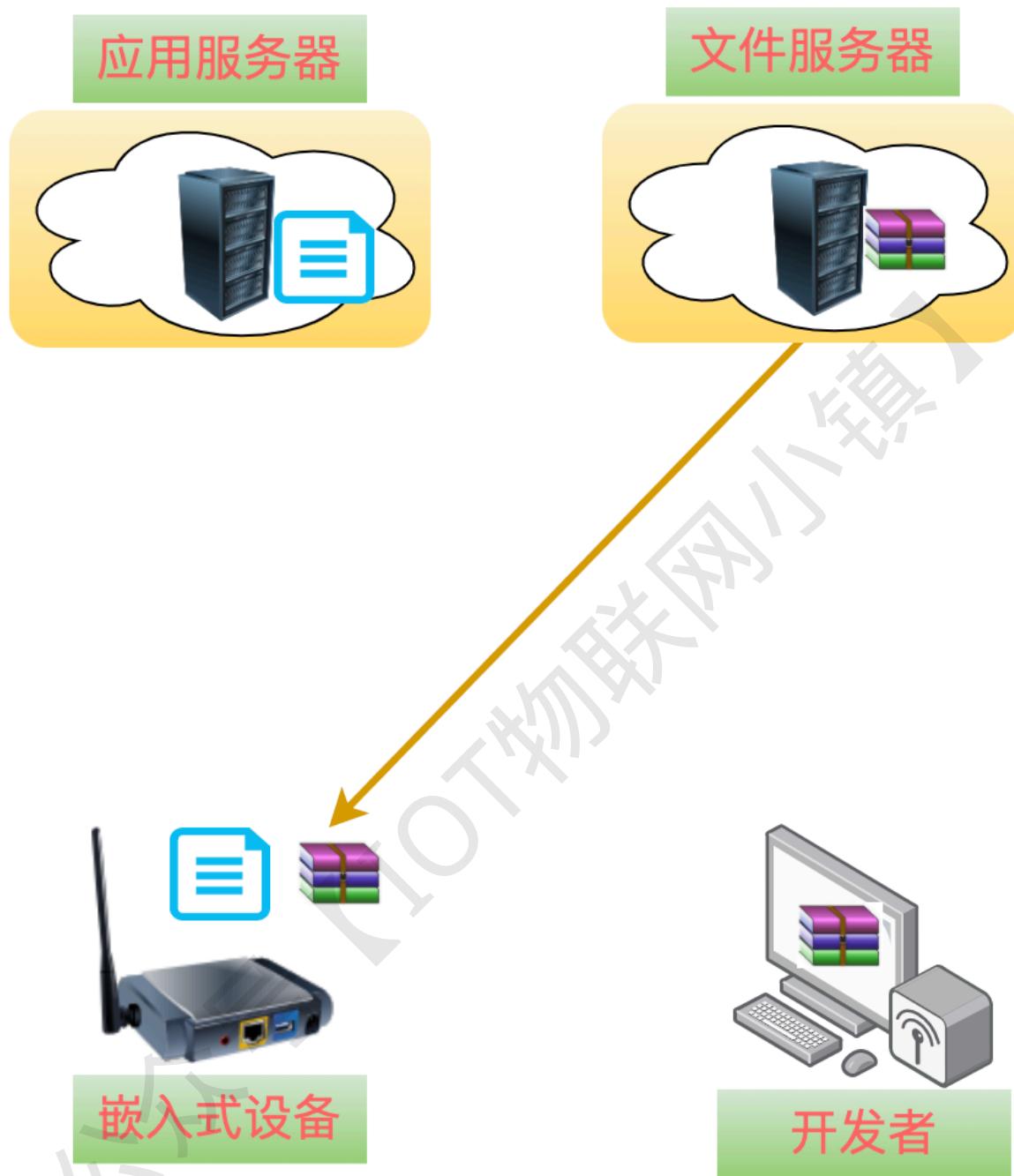
如果当前运行的版本已经是最新了，那就到此结束！

下载升级包

到了下载升级包的过程就简单了，你可以直接用 wget 等工具来下载，也可以利用 curl 库来手写下载代码。

公众号【IOT物联网小镇】

总之，你可以有一万种方式把我下载到设备中。



下载完成之后，有一件很重要的事情千万别忘了，那就是：[检查下载的升级包是否正确](#)！

还记得升级包描述文件中的 md5 字段吗？那就是我的[指纹](#)信息。

你需要首先计算一下下载的升级包的 md5 值，然后与[升级包描述文件](#)中的 md5 字段中的值进行比对，如果完全一致，那就放心大胆的开始解压、升级吧！

解压升级包

公众号【IOT物联网小镇】

欲知后事如何，请听下回分解！

----- End -----

Hi~您好，我是道哥，一枚嵌入式开发老兵。

这是我的[个人微信](#)，做个点赞之交也不错哦！



让知识流动起来，越分享越幸运！

[星标公众号](#)，能更快找到我！

[推荐阅读](#)

公众号【IOT物联网小镇】

- 【1】 C语言指针-从底层原理到花式技巧，用图文和代码帮你讲解透彻
- 【2】 一步步分析-如何用C实现面向对象编程
- 【3】 原来gdb的底层调试原理这么简单
- 【4】 内联汇编很可怕吗？看完这篇文章，终结它！
- 【5】 都说软件架构要分层、分模块，具体应该怎么做

公众号【IOT物联网小镇】