支付系统设计

账户体系是支付系统的基础，它的设计直接影响整个系统的特性。这里探讨如何针对电子商务系统的支付账户体系设计。我们从一些基本概念开始入手，了解怎么建模。

# 支付账户和登录账号

账户体系设计首先要区分两个概念，支付账户和登录账号。 这是两个不同业务领域的概念：**支付账户**指用户在支付系统中用于交易的资金所有者权益的凭证；**登录账号** 指用户在系统中的登录的凭证和个人信息。 一个用户可以有多个登录账户，一个登录账户可以有多个支付账户，比如零钱账户，储值卡账户等。 一般来说，支付账户不会在多个登录账户之间共用。如果没有特殊说明，下文中的账户，都默认指支付账户。

# 账户的设计需求

在支付系统中，账户的设置，主要是从如下几个方面来考虑：

**1、交易的需求**，比如检查账户是否被锁定、余额是否足够、是否有效等。

**2、记账的需求**，按照公司会计需求记录账户上的所有行为，包括支出、充值、转账等。

**3、对账的需求**，包括和支付渠道、商户、个人的对账需求，核对交易和账户余额是否正确。

**4、风控的需求**，如反洗钱、反欺诈等，都需要依赖于账户体系来提供核心数据。本文暂不分析这个内容，将在《支付风控》、《支付反洗钱》这两篇文章中详细分析

**5、信用的需求**，对用户、资产、商户等主体进行信用评估时，也需要依赖账户体系来提供的核心数据。本文也暂不分析这内容，将在《信用与支付》一文中分析。

这五个需求，按照其设计的优先级，也是从支付、记账、对账、风控来进行。 支付系统根据其发展所处的阶段，逐步将新增需求纳入设计中。

# 交易与账户

账户设置，一般是从交易开始的。 交易的实现必须有账户的支持，账户是交易的基本构成元素。 从支付系统的角度，交易中涉及到的资金流是资金从一个账户流向另一个账户。 发起交易的一方，被称之为交易主体，他可以是个人，也可以是一个机构。

资金从该主体所拥有的账户中流出。 而接收交易的一方，被称为交易对手，他也可以是个人，或者机构。 和第三方支付或者金融机构的交易不同，电商系统中，交易还会涉及到渠道。

由于电商系统本身并无清结算的资质，所有资金从交易主体到交易对手的账户的流动，在大部分情况下，并没有经过电商系统，而是由电商系统调用支付渠道提供的接口，由它来完成真正的支付过程。 当然，渠道也不是活雷锋，在这过程中，渠道要收取费用。

所以，在电商系统中，一次交易会涉及到三个账户： 交易主体账户、交易对手账户以及支付渠道账户。 如何在这三个账户中完成一次交易，我们将在后续的《交易和记账》一文中详细分析。

# 记账与账户

公司的会计需要对每一笔交易都要做详细的记录，即记账。 公司每天都产生大量的交易行为，为了便于管理和统计，一个简单的方法是对交易进行分类，比如食品、带宽、办公用品等等。 这个分类，按照公司的规模和业务复杂度，可以有一级，二级，三级或者更多级的结构，这被称之为会计科目。 记账时，除了交易明细，还需要在每个级别上对交易额进行汇总。

一般来说，一级科目上汇总称为总帐科目，而详细记录称为明细科目。 在电商系统中，由于涉及到的参与方较多，记账也相对复杂，但基本方法也是类似的。 电商的参与者可以分为商户、买家和渠道，对这三类参与者，都需要分别建立总帐账户和明细账户。

# 内部账户和外部账户

当用户使用银行卡来支付时，电商支付系统需要和银行对接，从用户银行卡所代表的账户上扣除资金。对接了银行，第三方支付等机构的电商支付系统，它需要连接到用户在这些机构的账户来执行扣款或者充值操作，这些账户或称为外部账户。对外部账户，支付系统只能记录账户在本系统的明细以及累计消费额，无法得知账户真正余额。 不少电商在玩零钱的概念，也就是让用户充值到零钱，使用的时候就直接从零钱中扣除。这就需要零钱账号。这是电商系统中自己设立的账号，所以也叫内部账号，可以知道账号的全部消费明细和余额。 当然，除了零钱账号，也可以有储值卡账号，信用账号等。

那问题来了，什么时候需要建立账户，比如优惠券，需要账户吗？ 一次消费的储值卡和可以充值的储值卡，需要建立账户吗？这里先埋个雷，后续介绍支付和记账时，给出答案。

# 收款账户和收单账户

当电商要对接银行时，往往都会被要求开设一个收款账户。用户通过这个银行来支付时，钱就被转到这个账户上。 对第三方支付也是一样。收款账户是开设在银行或者第三方支付这边的， 即渠道侧。 一般来说，渠道每天都可以提供这个账户的交易流水供电商对账用。 这样在电商这边，渠道就成为一个收单机构。 所以在电商这边，建立这个收款账户对应的对账用的收单账号，用来记录通过这个渠道进行的各项交易流水。

# 账户建模

说了这么多，目的是为了对账户建模。 账户模型是和公司业务密切相关的，公司不同规模，发展的不同阶段需要不同的模型。 账户建模本身包括三大核心模型：实体模型、账户模型和交易模型。 从交易模型中可以衍生出针对各个角色的账户流水，即明细模型，用于支持对账。

## 实体模型

实体模型和用户、商户模型有重叠的地方，这里专门针对支付而设置的各个实体属性。 一般来说，支付相关的实体模型需要包括如下的属性：

1、用户ID，一般直接映射到登录账户的ID；

2、是否允许执行支付；

3、支付密码；

4、用于设置或者重置支付密码的手机号；

5、用户设置或者重置支付密码的邮箱；

6、用户的安全等级，根据业务需要来设置。

## 账户模型

根据业务需要，可以设置多种账户，如支付账户、预付卡账户、代扣账户、零钱账户、结算账户等。 从类别上来说，这里的账户，一般指总账账户。一般来说电商系统中涉及的账户类型有：

* **虚拟币账号**：用户和使用奇点奇豆的商户都需要建立虚拟币账户。
* **代扣账号**： 用来支持订阅类型的定期代扣；
* **零钱账号**：即电商的内部账号，用户、商户、清算单位需要建立零钱账户
* **第三方支付账号**：用户在第三方支付机构建立的账户。
* **银行卡账号**：用户的银行卡信息，每个卡对应一个账户。
* **结算账号**：用来支持和第三方支付公司、银行进行结算用。 第三方支付需要为每个商户号建立结算账号；银行需要为借记卡、贷记卡分别建立结算账号（有必要吗？银行卡直连时使用）。
* **代扣代缴账户**：用来支持代扣税款业务。

对这些账户，需要设置如下属性： 基本属性，包括：

* **账户号**，或称为账户ID，一般是系统自动生成。特别注意的是，要事先约定好账户ID的规则。比如头三位用来表示账户类型，后几位用来表示账户编号等。务必保证根据账号号能够快速确定账户类型，并且保证账户号是不重复的。
* **账户名称**，一般是由用户自己设置的，显示用。
* **账户使用的货币类型**，注意虽然一张银行卡可以支持多个币种，实际在内部，还是针对每个币种建立独立的子账户。 涉及到多币种的账户，也可以采用类似的建模方案。
* **会计科目代码**，一般是一级会计科目的代码。

# 账户控制相关：

* 是否允许充值；
* 是否允许提现；
* 是否允许透支；
* 是否允许支付；
* 是否允许转账进入；
* 是否允许转账转出；
* 是否有安全保障；
* 是否激活；
* 是否冻结。

## 资金相关：

* 当前账户余额：等于可用余额+冻结余额；
* 当前账户可用余额；
* 当前账户冻结的余额。冻结余额指在账户上暂不能使用的额度。在支付的时候，往往是先冻结，商品出库后， 再实际执行扣款。

银行卡、第三方支付信息：

* 第三方实体的ID；
* 第三方账号，如银行卡号或者在第三方支付的open\_id等；
* 第三方的app\_id；
* 账号的失效日期，该账号什么时候失效。

注意，有些第三方信息是不能保存的，如用户的账号密码、信用卡的CV号等。 为了避免账户信息被爬库或者数据库信息意外泄露，一般还需要对敏感字段，如密码等，进行加密保存，甚至保存到另外的表中。 更进一步，为了避免账户信息被意外修改，还可以增加一个校验字段，在写入数据时设置该字段，在读取数据时做校验，一旦发现数据有问题，则关闭该账号。

## 交易模型

交易记录，交易流水，账户流水，交易台账，这三个容易混淆的概念，从数据上来说，却并不复杂，它们的核心是交易流水，账户流水是从账户视角的交易流水。那对一笔交易，涉及到的方方面面内容很多，有哪些需要记录的呢？考虑到交易记录将被用于风控和信用分析，能收集到的信息是越全面越好。

* 流水号：每一笔交易的流水号都不一样。需要根据业务情况详细设计流水号。这个号往往也是对交易表做分表分库的依据。
* 交易记录创建时间；
* 交易记录最后修改时间；
* 会计科目代码
* 关联的订单号，由商户提供；
* 订单名称、描述、关联的地址等信息；
* 费用信息，包括： 结算货币类型、原始费用、实际费用等；
* 交易主体信息，记录主体ID、类型、名字、账号、账号类型、使用的IP地址、手机号、平台、通知邮箱、当前位置等。 这些信息虽然可以从主体表中获取，但考虑主体表信息随时会被修改，所以这里需要记录详细的各原始信息。
* 交易对手信息，记录对手主体的ID，类型，名字，账号，账号类型，手机号，平台，通知邮箱等。
* 交易渠道信息，记录所使用的交易渠道的实体id，渠道账户，渠道执行支付的时间、渠道侧返回的订单号等。如果有错误发生，还需要记录从渠道接收到的错误信息和错误码。

可以说，对账是支付系统最头疼的事情。每一笔交易，都要做到各参与者的记录能够吻合，没有偏差。对账系统的工作，是发现有差异的记录，即轧帐；然后通过人工或者自动的方式，解决这些差异，即平帐。

对电商系统来说，每一笔交易，在所有相关主体侧都要能对得上：

* 交易主体，如果发起人是个人，必须能够从个人交易历史记录中找到这笔交易。但大部分人不会保留电子记录，所以一般是提供可以下载的账单或交易记录，让用户自己对去。
* 交易对手，一般是商户。商户侧对账处理同用户侧，也仅仅提供对账单。
* 交易渠道侧，这是对账的重点，一是核实交易流水，二是核实交易佣金，毕竟是租用人家通道做结算的。

那有哪些记录需要对账？ 目前主要是两个：一个是交易记录；一个是退款记录。

# 对账处理流程

一般来说，对账流程涉及到如下步骤： 渠道对账单下载、本地交易记录准备、轧账、平账。

## 渠道对账单下载

银行，第三方支付，银联等，基本都会提供对账单下载的功能。不过也有少数工作做不到位或者太到位的银行，只提供账单查询后台，不提供对账单下载功能。

对开发人员来说，这里有几个坑：

* 对账单格式不一。文本，XML，csv的都有。为了后续能够统一处理，在账单下载完成后，需要进行标准化处理。
* 下载方式不一，HTTP，HTTPS，FTP的，都有。下载程序需要按照渠道的协议来处理。
* 下载时间不一，一般是凌晨1点后，到中午12才能用的也有。如果在预定的时间取不到数据，需要注意重试读取。
* 稳定性差。FTP服务器出问题那是常有的事。渠道侧解决方案往往就是重启。所以重试机制是必要的。

看一下第三方支付的对账单情况：

银行直连的对账情况：

技术选型上，HTTP(S)用apache httpclient即可实现链接池和断点续传， FTP也可以使用Apache Commons Net API。 但不管是哪一个，都需要设置重试次数和链接超时间。重试次数和间隔的设置需要小心，重试太频繁，容易把服务器打死.；时间间隔太大，又会阻塞后续处理步骤。5～10分钟是一个合适的重试间隔区间。

链接超时指在服务器出现问题时，连接在指定时间内获取不到数据即自动断开。这个很容易被忽略。我们有一次系统出问题，是渠道侧的FTP假死后重启，导致我们的客户端挂住，一直在等待重新链接。

渠道对账单标准化：

找个例子大家看看， 比如微信的对账单，他是csv格式的，包括如下信息：

1. 交易时间：这是在微信侧的支付完成的时间。 这个时间会成为一个陷阱。
2. 公众账号ID，商户号,子商户号,设备号： 这些信息需要做验证，确保是自己的单子，不要让微信把老王家的单子也给发过来了；
3. 微信订单号,商户订单号： 这两个是对单的核心。前者是微信侧产生的订单号，在微信支付接口返回值中有。但是万一收不到这个返回值，那在本地记录中可能就空了。 后者是我们发送给微信的订单号，一般用这个来做对单依据。两边的数据中都会有这个值。
4. 用户标识,交易类型,交易状态,付款银行,货币种类,总金额,企业红包金额： 这几个就是对单的核心字段，必须确保双方是一致的。
5. 商品名称,商户数据包,手续费,费率：这些是可选验证。

而某宝的对账单，是文本格式的，用空格隔开。他们家的就简单很多，只有商户订单号，交易流水号，交易时间，支付时间，付款方，交易金额，交易类型，交易状态这些字段。

由于每个渠道的账单格式都不尽相同， 在得到账单后，下一步是对账单做标准化处理，这样轧帐以及后续工作就可以统一处理了。 标准化后的账单数据可以放在文件系统或者数据库中。这取决于交易数据量。每天百万以上的量，还是使用文件系统，比较合适。数据库操作相对比较慢，也浪费资源。

基于文件系统的标准化涉及如下内容：

* **文件格式标准化：**统一使用csv或者json或者xml格式。如果是使用hadoop或者spark来对账，使用csv是个不错的选择。
* **文件存储统一化：**文件目录，文件名都需要遵循统一命名规范。

为了加快处理速度，我们使用hdfs作为文件系统，有利于后续的对账的处理。

## 本地交易记录准备

本地交易记录的准备，总的来说有如下方法: – 啥都不做，直接用原始数据。鉴于大部分系统使用的是mysql，这也意味着在MySQL上做对账。对账时需要大量的数据查找工作，必然会影响线上业务。在数据规模较大，比如超过100万时，就不太合适了。

* 当然，还有一个选择是使用备库来执行对账，这样既简单，也不影响线上业务。这是典型的空间换时间的做法。
* 如果业务大到需要分表分库才能处理，那对账数据准备也不一样。使用分库也不现实，因为分库一般是按照主体id，而不是渠道id，来分库，这样对账就需要在多个库上进行，效率反而降低了。而对分表分库建立从库也非常耗费资源。这种情况下，需要同步一份数据到(hdfs)文件系统中，或者NOSQL数据库上。

由于交易记录是支付系统核心数据，有大量的应用，如信用、风控等，都需要交易记录数据。这些应用对交易记录的需求还不完全一致，为了提升性能， 交易记录会使用异步的方式来将数据投递给使用方。 交易记录在入库时，投递消息到消息系统中。使用方监听这个消息，一旦收到新消息，则从交易记录库中查询数据，获取数据并更新到库中。关于此类数据同步的文章不少，这里就不详细介绍。

## 轧帐

轧帐是按照客户订单号来比较本地交易记录和渠道交易记录是否一致。从算法角度，是计算两个数组的差异。在单机运行时，可以采用的算法不少，这里不详细介绍。 我们推荐采用mapreduce来轧帐，这有个优势，可以按照订单号将渠道提供的记录和本地记录shuffle到同一个reduce处理上，这样就可以很容易进行数据比对。 轧帐中最大的坑，莫过于切分点的问题。

比如以整0点为切分点，那存在一个问题，本地23:59发起的交易，到了渠道侧，可能会在00:01处理，这一笔交易变成第二天的帐了。实际处理中，一笔交易在渠道侧处理，花上几分钟都有可能。 对于切分点附近无法确认的帐，做一个时间窗，在时间窗内的数据，留待第二天对账时继续处理。

## 平帐

发现两边不一致的数据，那应该如何处理？数据量不大时，记录起来，人工甄别就行。但如果数据量很大，每天上千条，人工处理就成本太高了。这个没有统一的处理方法，需要根据有问题的数据，做个分析，然后做自动处理。 针对交易记录的对账的处理，主要有如下情况：

* 本地未支付，支付渠道已支付。这主要是本地未正确接收到渠道下发的异步通知导致。 一般处理是将本地状态修改为已支付，并做响应的后续处理，比如通知业务方等。
* 本地已支付，支付渠道已支付，但是金额不同，这个需要人工核查。
* 本地已支付，但是支付渠道中无记录；或者本地无记录，支付渠道有记录。在排除跨日因素外，这种情况非常少见，需要了解具体原因后做处理。

针对退款的对账处理，主要有如下情况：

* 本地未退款，支付渠道已退款，则以支付渠道为准，修改本地为已退款状态，并出发后续处理。
* 本地已退款、支付渠道已退款，但是金额不同，需要人工核查；
* 本地已退款，但是支付渠道无记录；或者支付渠道有记录，但是本地没有。 在排除跨日因素外， 这种情况非常少见，需要了解具体原因后做处理。

总之，对账工作，即复杂也不复杂。需要细心，对业务要有深入的了解，并选择合适的架构。

这一期，回到支付系统的核心业务，即支付。每个电商公司的支付系统都已经或多或少的实现了交易核心功能，可也都是一直在改进，总是不断的有新的需求冒出来。所以这一期开始，我们梳理一下：到底有哪些支付方式？每种支付方式都是怎么运作的？

# 支付和交易

说到支付就不得不提交易。这两个概念在不同公司中是不一样的。我们的定义是，交易是生成订单；支付是对订单进行付款。 订单生成过程我们以后另开话题来说。这一次重点介绍支付。而就支付行为来说，我们碰到的大部分都是单次支付，其次还有转账和退款。在苹果推出订阅支付后，国内支付宝等也在陆续跟进。 单次支付是我们用的最多的支付方式了，即一次结清所有款项。把单次支付走通了，其他支付方式也容易处理。 本期重点介绍单次支付。

## 银行卡支付

先说大家比较熟悉的银行卡支付，它分为线上支付和线下支付两种形式。线下支付就是通常说的POS收单，这里不介绍这个内容。对线上支付，按照卡的类别，分为贷记卡支付，也叫motopay、ePOS，即信用卡支付；和借记卡支付。按照支付形态，又分为认证支付、网银支付、快捷支付几种形态。银行卡网银支付要求银行卡必须开通在线支付功能，而快捷支付并不需要开通在线支付功能。主要利用支付验证要素（卡号、密码、手机号、CVN2、CVV2等），结合安全认证（例如短信验证码），让持卡人完成互联网支付。

### 认证支付

指用户在绑卡时，将卡信息提供给电商。这样在支付时，用户无需再输入这些信息，由电商在服务器侧保留用户的账户信息，比如身份证号，卡号，手机号。在用户支付时，无需再输入这些内容，最多就提供个密码或者校验码，就可以完成支付。这基本不会打断用户的使用体验，所以也是电商喜欢的支付方式。但认证支付最让人诟病的就是安全性。一方面需要向电商暴露个人信息，一旦被窃取，资金就容易被盗走。还有在手机上执行支付，一旦手机丢失，窃取者就可以轻而易举的使用或者转移资金。

### 快捷支付

快捷支付和认证支付类似，不同点在于绑卡之后，有些银行接口会返回token，后续使用token来作为支付凭证，无需提供卡号信息，这样电商也不需要本地保留卡号了。目前主要是银联有提供token接口。

### 网银支付

相对来说，网银支付要安全很多。网银支付是由银联或者银行提供支付界面，用户必须在页面上输入卡号，密码等验证信息才可以执行支付。大部分银行还要求用户使用U盾或者其它安全硬件。但安全和易用永远是个矛盾。网银使用会打断用户体验，增加用户使用难度。对使用硬件加密的支付，不可能天天带着U盘跑。另外网银主要用在web端，在手机端，嵌入网银页面，还是比较难看的

## 支付流程

走一个具体的例子看看吧。比如用户在电商系统中买了200块钱的东西，然后通过浦发银行卡做结算，用的是快捷支付。这个过程是：

用户在交易界面上，提交订单到交易系统中； 交易系统确认订单无误后，请求支付系统进行结算。这是在交易系统做的，后面工作就进入支付系统。

用户被引导到收银台页面， 让用户确认交易金额，选择支付方式，调用支付系统接口。

支付系统接收到支付请求，验证请求的各个字段是否有问题，确认无误后，调用支付网关执行支付。

支付网关请求浦发银行的快捷支付接口执行支付。

支付网关接收到支付结果报文后，对结果报文做解析，获取结果，并将结果告知交易系统。这可以通过URL或者RPC调用来实现。

商城系统收到支付结果后，开始执行后续操作。如果是支付成功，则开始准备出库。这一步在交易系统中处理，这里不做介绍。

网银支付，和快捷相比，就在第4步，插入一个步骤，将用户导航到网银页面输入支付信息，后续步骤是一样的。在资金流上也是相同的。 而在第五步获取返回结果上，一般银行就直接同步返回，银联是分为同步和异步返回。同步告知操作成功或者失败，异步告知扣款成功或者失败。同步操作和异步操作都需要调用方提供一个回调的URL地址，银联会将参数附加在这个地址上。通过解析这些参数可以得到执行结果。异步操作一般有2-3秒的延迟，取决于网络，以及该交易处理的复杂度。

### 资金流

上一节说的是支付的信息流，那资金流应该是怎么走的？ 在第三步，会触发资金流。资金从用户个人账户上转移到电商公司的账户。当然，银行也不是活雷锋，这一笔交易是要收手续费的。资金是实时到账的，手续费一般是按月结算。有按交易笔数计费的，但大部分还是按照交易金额来收费。

同行快捷支付是比较简单的场景，让我们来逐步增加难度。如果支付系统没有对接浦发银行，那对浦发卡，就得走其它支付方式：银联或者第三方支付。

先说银联快捷。银联提供的多种接入方式，常说的快捷支付，在银联文档中叫商户侧开通token接口。通过这个接口，可以实现同行和跨行资金结算。不管收款行是浦发还是其它行，都可以完成结算。对本地和用户来说，体验是一样的。而在银联侧，后台资金流处理却不一样。了解这个资金流，有助于在异常情况下，了解资金到底跑到哪里了。

如果收款行也是浦发银行，银联发报文给浦发，浦发使用内部系统完成两个账户间的转帐，即时完成。

如果收款行是他行，比如工行。银联发指令给浦发和工行，分别完成各自账户上资金余额的增减，对个人和电商来说，这笔资金算是落地了。但实际资金流并不是立即发生。银联会在半夜做清结算后处理这笔资金。这个过程就是金融机构之间的清结算了，一般不需要关注。

如果使用的是第三方支付，对用户来说，处理的流程和银联一样。但资金流会不一样。 第三方支付在浦发和工行一般都会有落地的托管资金。 发生交易后，一般来说不会产生跨行资金流动。用户在浦发行的钱会被结算到第三方支付在浦发行的托管账户，而在工行的钱，会由第三方支付在工行的账户打到客户账户上。 这就降低了跨行资金流动成本。

目前国内主要银行都提供快捷和直联的接口。对电商来说，要对接哪些银行是个需要考虑的问题。怎么对接银行，渠道和第三方支付。

## 银联Token支付

一般来说，大部分银行都提供直联和网银接口，但不需要直接对接所有银行。银联和第三方支付也提供直联接口，可以直接对接国内主要银行。也不是所有银行都被银联支持，这和银联签约的接口有关，需要在对接时咨询银联。从我们使用情况看， 浦发借记卡、邮储银行卡是不支持的。 另外 交行、平安（含原深发）、上海银行、浦发、北京银行，上述银行卡需通过 这个地址 开通银联在线支付业务。

### 对接银行

大部分银行提供的银行卡支付接口，借记卡支付和贷记卡支付是不一样的。但也有几个好心的银行，可以用一套接口同时开通借记卡和贷记卡。点名赞一下这些银行： 宇宙第一大行工商银行和建设银行。其他同学对接中如果也发现借记卡和贷记卡用一个接口的，也请及时告知。 作为国内最保守的软件团队，和银行对接时务必做好足够的准备。在商务谈判完成、拿到银行的接口文档后，需要考虑两个问题：专线问题、加密问题。

### 专线问题

首先是专线问题。 大部分银行对接是需要专线的。 与银行沟通的时候，注意收集如下信息：

* 专线类型： MSTP类型或者SDH类型。
* 专线接入点：目前国内主要是联通、电信。
* 封装类型： HDLC或者PPP
* 专线代宽：默认是2M

前置机IP，这个需要在银行侧和电商侧进行配置。 专线其实是在银行和电商之间建立一个局域网，需要双方分配通讯IP。 其实这两组IP都是NAT后的IP，银行分配给我们的是电商真实的前置机IP经过最外端的网络防火墙转换后的IP段，后者也是对方的真实前置机IP经过转换后的IP段。 出于安全考虑，双方都不会将真实IP暴露出去，所以要NAT。

接入地址：即电商这边机房的地址。

这些专业名词，可以自己检索，太专业了，其实我也不懂。从可靠性角度考虑，一般建议从联通、电信各拉一条线路出来。一旦有一个线路出问题了，也不会导致所有交易被终止了。不需要专线的银行接口有：浦发、工行、交行信用卡等。 需要专线的有中行、农行、建行等。一般专线需要1个月左右的时间，包括银行侧的申请、施工时间。

### 加密问题

其次是加密问题。部分银行，如中行，前置要求使用加密机。此处加密机的常用功能有三方面：

* MAC加密（完整性）；
* 支付会话\密码加密（安全性）；
* 密钥交换加密（防截取）。

对开发来说，加密机的主要作用，是让黑客都无法从内存中看到密码。 不是做广告，国内对接银行一般就用江南天安的加密机了

### 对接银联

对接银联比对接银行简单， 不需要专线，不需要加密机。 不过需要获取ADSS认证。 银联最近在推Token接口，有两套接口，一套是银联侧开通，一套是商户侧开通。前者类似网银支付，后者类似快捷支付。 务必要求接入后者接口啊。基本上读完接口文档就知道怎么写代码了。

在上一篇 支付系统之银行卡支付中，挖了个坑，就是关于绑卡的坑。 在用户使用银行卡做支付之前，首先需要完成绑卡的操作。怎么实现绑卡，怎么验证用户绑的是自己的而不是隔壁老王的卡，这就是本期的重点。

为什么要求用户绑卡？这和快捷支付有关。参见上一篇文章的分析，绑卡是将用户卡信息提供给电商，以后电商就用这个信息去银行完成支付。绑卡实际上是一个授权，让用户允许商家自动从他的账户上扣除资金。所以绑卡也叫签约，用户和银行，商家的三方签订的支付合约。 但我们知道，绑卡对用户和商户来说都存在巨大风险。

如果说用户绑卡是图省事，那商户为什么要做这个事？首先当然是提升用户体验了，让用户花钱更容易。其次，提升支付成功率。使用网银支付成功率在20%左右，银联直联成功率一般在50%左右，银行卡直联可以提升到70%左右。这是相当可观的数据。所以，当你看到绑卡送洗衣粉之类做法时，不需要担心商家会不会赔本。

怎么绑卡？我们知道对接银行有两种途径，直接对接银行接口和通过银联来间接对接。这两种情况下绑卡处理也不同。

## 绑卡场景

直观的，电商网站会在用户后台提供一个绑卡的入口，让用户直接绑卡。以支付宝绑卡流程为例，我们可以体验下：

这里有如下要点：

* 只能绑自己的卡，这主要从安全角度考虑。
* 需要用户在银行侧预留的手机号进行短信验证。但不是所有银行都需要。这个时候，为了统一处理，可以考虑自己发验证短信。

对这个入口不要指望太多，更多的用户是在支付中绑卡。也就是提交订单后，发现没有银行卡了，就开始绑卡。 和纯绑卡流程不同的是，最后一步，绑卡成功后，一般都同时完成支付。有些渠道会提供绑卡并支付的接口，减少交互次数。

## 绑卡流程

先介绍比较简单的银联直联绑卡。为了保证卡的安全，绑卡有这些前置需求： 1、用户必须已经绑定了手机号。该手机号用于修改支付密码；

2、用户需设置了支付密码。支付密码不同于登录密码。

针对用户不同状态，绑卡流程上有区别。当然，绑卡是安全操作，要求用户必须登录到系统中。为了避免和服务器端的交互被劫持，所有操作必须在安全链接中进行，即使用https。当用户开始绑卡时，执行如下流程：

1. 检查用户是否有手机号。没有则进入设置手机号流程。
2. 检查用户是否设置支付密码。如果已经设置，则需要用户输入密码。确认后开始绑卡。否则，也是先进去绑卡后设置密码。
3. 用户输入卡号，系统根据卡号判断卡的发卡行，并显示给用户。有些实现，如微信支付，会提供扫卡识码功能。
4. 用户输入银行预留手机。对于没有绑过卡的用户，需要用户提供真实姓名和身份证号。对于信用卡，还需要输入cv码和有效期。这一步，卡的信息都收集全了。
5. 调用银行绑卡验证接口进行绑卡。这里有一个四要素验证的概念。由于国内要求实名制，所有银行卡都是实名办理的，所以银行可以验证姓名，身份证号，银行卡号和手机号是不是一致的，如果没问题，则会发短信到手机上。
6. 用户输入短信验证码并确认绑卡，服务器端将用户实名信息以及短信验证码组合形成报文，发送给银行，执行签约操作。银行侧签约成功后，返回签约号给商户。

## 卡bin

这里有个问题，如何根据卡号判断发卡行？这就需要卡bin。 BIN号即银行标识代码的英文缩写。BIN由6位数字表示，出现在卡号的前6位，由国际标准化组织（ISO）分配给各从事跨行转接交换的银行卡组织。银行卡的卡号是标识发卡机构和持卡人信息的号码，由以下三部分组成：发卡行标识代码（BIN号）、发卡行自定义位、校验码。

目前，国内的 银行卡 按照数字打头的不同分别归属于不同的银行卡组织，其中以BIN号“4”字打头的银行卡属于VISA卡组织，以“5”字打头的属于MASTERCARD卡组织，以“9”字和“62”、“60”打头的属于中国银联，而“62”、“60”打头的银联卡是符合国际标准的银联 标准卡 ，可以在国外使用，这也是中国银联近几年来主要发行的银行卡片。 大部分银行卡号前6位即可确定发卡行和卡类型，但也有非标卡需要6-10位才可以判断出来。需要维护一个卡bin库。附件是一个比较完整的卡bin库， csv格式的。

## 短信和身份验证

一般绑卡操作第五步需要银行下发短信验证码。 短信验证的接口，不同银行还不一样。有些银行是短信和身份验证一起做了；有些银行是可以配置身份验证是否同时发短信。还有些比较奇葩的机构，比如某联，接口中让你传身份信息，但实际上没传也是可以的，也不验证身份信息到底对不对。这在对接渠道时需要特别注意。

此类接口一般包含如下内容：

* **版本号**：当前接口的版本号；
* **编码方式**： 默认都是UTF-8，指传输的内容的编码方式；
* **签名和签名方法**： 生成报文的签名。 不是所有的字段都需要放到签名中，文档中会说明哪些字段需要签名；
* **签名算法**：生成签名的算法，RSA, RSA128， MD5等。
* **商户代码**：在渠道侧注册的商户号。
* **商户订单号**：即发送给渠道的订单号；
* **发送时间**：该请求送出的时间。
* **账号和账号类型**： 银行卡、存折、IC卡等支持的账号类型以及对应的账号；
* **卡的加密信息**：如信用卡的CVN2，有效期等。
* **开户行信息**：开户行所在地以及名称；大部分是不需要的。
* **身份证件类型和身份证号**： 可以用于实名验证的证件，指 身份证、军官证、护照、回乡证、台胞证、警官证、士兵证等。不同银行可以支持的证件类型不一样，这也不是问题。大部分就是身份证了。
* **姓名**：真实姓名，必须和身份证一致；
* **手机号**：在所在银行注册的手机号。

系统会返回上述数据的验证结果。如果验证通过，则会发短信。但这不是所有的渠道都是这样。哪些字段会参与验证、需不需要发短信，需要注意看接口文档。

绑卡接口

绑卡接口和发短信接口类似，还需要将用户的卡号，身份证等信息传递过去。在绑卡成功后，会返回一个签约号。这个签约号是后续调用支付，解约等接口所必须的。 这里有个问题，已经绑卡的用户，调用绑卡签约接口再绑一次，会出现什么情况？这个和银行实现有关。 大部分银行，如农业、浦发、建行等，对绑卡签约接口调用，会首先验证身份信息，如果验证不通过，则不执行后续操作。验证通过后，再检查这个卡在该商户下是否已经绑过了， 如果没有绑过，则执行绑卡，否则会提示卡已经绑定过了，不能重复签约。 但工行的实现不一样，他是首先验证这个卡是不是已经绑过了，如果已经绑卡，则不继续验证身份信息。 总之，银行都不支持重复绑卡。

## 银联绑卡

银联直联绑卡和银行绑卡类似，但是得注意验证接口，仅验证卡号和姓名，不验证身份证号和手机号。这导致第5步无法正常进行。银联只有到第六步执行绑卡时才做身份验证。 所以在处理上，还需要做一些调整，来确保和银行的流程的一致。 一种处理方法是，对银联，在第五步就开始调用银联接口执行绑卡操作，但是在本地标记为预绑卡状态；商户侧发送短信验证码，验证通过后，才将状态设置为绑卡成功。

银联网银绑卡处理起来比较麻烦。用户在电商页面上输入卡号，然后被导航到银联页面上去完成绑卡操作，成功后，银联返回一个token作为签约号，用于支持后续操作。这问题就来了，用户可以在银联页面上绑定一个别人的卡，而电商侧是无法知道这个卡的情况的。所以这种方式尽量不要用。

## 实名认证

绑卡操作有个不错的副产品，就是实名认证。常说的二要素，三要素，四要素认证，可以通过这个操作完成。 二要素指姓名和身份证号，三要素加上银行卡号，四要素则加上手机号。看起来，似乎银行都应该支持四要素验证，但大部分银行接口仅支持三要素，毕竟手机号还是非常容易变。 当然，实名认证，也就是二要素认证，是应用最多的认证了。国内唯一的库是在公安部这，由NCIIC负责对外提供接口。可以提供如下功能：

* **简项核查**：返回“一致”“不一致”“库中无此号”
* **返照核查**：返回“一致+网纹照片”“不一致”“库中无此号”
* **人像核查**：返回“同一人”“不同人”“库中无此号”

官方接口收费是 5元/条。 市面上主要的第三方服务提供商有国政通（简项、返照）、诺证通（简项）、IDface（三接口）等，收费简项核查：0.5~2.0元、返照核查为0.8~2.1元、 人像核查2.0~8.0元不等。一般都和访问量有关，量大从优。

当然，这里也要注意，涉密人员是没法查到相关信息的。 性能上， XX通一般在200ms内即可返回结果，普通商用应该是没问题的。 有些公司还会额外提供四要素接口，以XX通为例，它号称支持大部分银行卡的四要素认证。但是实现上有点儿懵，居然是实时请求银行的接口，这就导致接口延迟非常高，1秒以上的占大部分，甚至10秒以上的都不少见，基本无法商用。这种情况下，还不如直接上银联。

应用内支付指使用手机操作系统自带的支付功能来支持支付。目前国内主要的应用内支付有 Google Pay、Apple Pay、小米支付、华为支付等。 其中Apple Pay是典型的一个应用内支付，Android平台的各种支付也一般是沿用Apple Pay的设计。

为什么要IAP

相对来说，应用内支付的用户体验，和微信支付、支付宝相比，还是有一定差距的，但是为什么要开发应用内支付呢？ 这个和苹果的AppStore的审核政策有关。 在官方的 (App Store Review Guidelines) 中， 有如下几条意见：

1.2 Apps utilizing a system other than the In-App Purchase API (IAP) to purchase content, functionality, or services in an App will be rejected.

在 App 内使用非 IAP 的系统来购买内容、功能或服务将被拒绝。

11.3 Apps using IAP to purchase physical goods or goods and services used outside of the App will be rejected.

IAP 购买实物或者应用外的商品或服务将会被拒绝；

11.4 Apps that use IAP to purchase credits or other currencies must consume those credits within the App

通过 IAP 购买的积分或者其他货币必须只在 App 内使用。

这问题就来了，如果要购买的服务，即在IOS内使用，也在Android等IOS系统外使用， 那应该是使用规则11.2或者规则11.3来执行？ 比如说视频网站，视频既可以在IOS上看，也可以在Android上看，那是否是需要通过IAP来购买？ 苹果公司在这一点上采取模糊的策略。 爱奇艺、腾讯视频，在IOS上购买会员，只能用IAP支付。这就和苹果公司的审核有关。

## IAP支付流程

一般IAP支付的开发流程，首先需要一些准备工作，包括：

1. 在developer.apple.com上配置一个App ID，使用该ID生成和安装相应的Provisioning Profile文件。
2. 登录到iTunes Connect，使用App ID创建一个新的应用，在该应用中，创建应用内付费项目，设置好价格和Product ID以及购买介绍和截图。
3. 添加一个用于在sandbox付费的测试用户，填写相关的税务，银行，联系人信息。

完成这些准备工作后，既可以进入正式的开发，开发代码我们这里就不说了，流程如下：

1. 用户选择要购买的内容并点击购买按钮；
2. 用户通过App Store账户验证
3. 苹果服务器验证用户请求
4. 苹果服务器从用户帐号扣款
5. 苹果向用户返回购买成功信息
6. 软件接收并显示用户购买信息

老司机都能看出来，这里有好多好多的坑。

用户访问AppStore时使用的是Apple的账号，不是应用系统的账号。 也就是说，我们并不知道到底是谁在购买这个内容。 比如在应用中有两个账号A和B，用A账号登录后，上IAP买了东西，然后用B账号来登录，也上IAP买东西， 这两次购买，用的是同一个Apple账号。苹果也不会告诉你，到底是哪个账号付了钱。 账号坑在单次购买中还没什么问题，但碰到订阅的情况，得好好处理下。在订阅章节中会详细说明。从上述流程可以看出，苹果服务器都是和客户端打交道的，这里面似乎没有应用服务器什么事情。 只有在客户端接收到苹果返回信息后，才可以把这个信息转发给应用服务器。 如果用户一直不打开手机上的应用，那应用服务器就一直收不到通知了。 好在后来苹果提供了一个验证功能，应用服务器可以把接收到的返回信息（加密后的字符串）发送给苹果服务器来验证和解密。

## IAP订阅

IAP Subscription又是一个大坑。 官方的文档在这里。内容不多，没有说明的东西却很多。

### 续费周期的计算

IAP主要提供给周期性订阅的音乐、电子书等内容使用。 一般就按月来计算周期。苹果是以自然月来算权益周期。比如在1月3号买了权益，到2月3号，这个权益就过期啦，需要在此之前完成续费。 那问题来了，1月31号买的权益，到几号过期？以自然月算，这个权益会在3月1日前到期，如果2月份，3月份都续费了，到4月份，也是享受到4月30日了。

### 自动续费

应用开发应该不需要关心续费的细节。苹果会做自动处理。在权益到期前10天，苹果检查用户账户是否可以扣款，商品价格是否有变动。在权益到期前24小时，苹果开始扣款，如果失败，会多次重试，直到成功。问题来了，这个重试，会延续到用户权益过期后一小段时间，苹果没有说这段时间该算是有权益还是没有，但开发人员需要注意应该如何处理。

### 免费试用

免费试用不是强制需求，但这有利于用户判断是否值得购买这个物品。免费试用期是在itunes connect中设置。 当用户第一次购买这个东西的时候， 客户端接收到的Receipt中包含免费试用信息。在免费期快到的时候，苹果发起第一次扣款。整个过程和自动续费类似，唯一区别是第一个月是免费的。

### Receipt 验证

客户端接收到 Receipt 之后，需要提交到服务器端进行处理，开通权益。 这就来了个问题：Receipt应该在客户端还是服务器端解析？当然需要在服务器端处理，这样可以防止越狱后的一些插件，如IAP Cracker、IAP Free等伪造交易凭证，欺骗苹果服务器，开通权益。 此外，还需注意，客户端和服务器端之间需通过HTTPS以及参数签名等方式来确保通讯安全。 服务器端接收到Receipt之后，首先验证请求的有效性， 然后将Receipt发送到苹果服务器上进行验证和解析。 接收到苹果处理结果后， 将Receipt中的user\_id, product\_id、purchase\_date、transaction\_id等做验证和处理。

## IAP破解和防御

既然Iap的验证主要是在苹果服务器端和手机客户端进行，并且是使用域名。这简直是为攻击打开了一扇大门，而不仅仅是漏洞。 早期的IAP内购解锁工具IAP cracker对IAP的破解比较简单粗暴。写过IAP程序的人都知道， 程序中基本都是用transactionState来判断交易是否成功。

transactionState 有四个状态：

1. SKPaymentTransactionStatePurchasing
2. SKPaymentTransactionStatePurchased
3. SKPaymentTransactionStateFailed
4. SKPaymentTransactionStateRestored

SKPaymentTransactionStatePurchased 表示购买成功了。 只要修改这个变量值，如果客户端应用直接根据交易状态来处理业务流程，那就会收到这个假的交易成功信息，接下来用户就能不花钱得到所买的物品。这个过程，甚至都不需要接入网络。

另一个工具IAPfree功能更强大，安装使用也复杂很多。它是通过修改DNS，让客户端访问黑客提供的服务器来取代访问苹果服务器，实现所谓的MITM中间人攻击。当用户在客户端触发购买流程时，会被引导到伪装的苹果服务器上，不扣款而直接返回扣款成功收据。用户不需要支付任何资金，客户端能够拿到完整的收据。如果是在客户端处理收据验证也没有任何问题。为了避免用户所使用的设备被封，这些软件甚至可以提供伪造UDID的功能。 为此，苹果特别说明，一定要在服务器端验证用户购买信息，验证内容包括收据签名，证书，产家信息等，确保收据无误后，才能授予权益。如果发现有诈，则将用户拉黑。

### 两套账户体系

苹果支付的账户体系,当然是以apple id为基础的，它允许用户在多台设备上共用一个账户。一台设备上，一般只有一个激活账户。但对应用系统来说，大部分是允许多个账号登陆的。这对续费来说就是个大问题。 用户以账户a登录后，发起续费，获得权益。然后以账号B登录了，显然，A的权益不会衍生给B。过几天A开始续费了，续费之后，切换到B账号登录，客户端在B账号登录时得到续费的收据并发送给应用服务器。那这算是谁的续费请求？当然是A的。在这个apple id发起的续费请求，所有的收据都会有一个相同的原始交易号original transaction Id。在用户发起订阅时，需要记录这个id和账号的关系，每次续费，需要在解析收据后，根据原始交易号从这里获取真正的充值账户，不能从客户端提交的用户id作为凭据。

还是这个坑，如果在账户b登录后也发起订阅请求，会怎么样？这个调用将会失败，所以需要阻止用户发起这样的请求。或者设置多个产品副本来让用户购买。

分成，定价和国际化

在iTunes中的给的产品定价必须是税前的，苹果和商家的分成，也是按税前算。商家给出在一个主要销售国家和地区（比如国内的基本就是中国了）的价格，即基准价格。在其他地区的销售价格，苹果会自动根据当前的汇率来换算成当地的货币。当然，也可以自己修改设定在这些国家或者地区的当地价格。目前是支持到155个国家。还要特别注意版权问题。

基准价格调整，如果是往高了调整， 则在用户下一次续费时，需要用户确认。如果往低了调，那就不需要用户确认，直接扣款了。

苹果对商家的产品价格体系有分组（Group）的概念，同国内说的价格体系，比如白金会员、黄金会员、贵宾等，在同一个Group里面，用户只能选择一个档，比如用户要么是白金要么是黄金会员，不会同时是。

在同一个分组中，如果用户订阅时间超过一年（365天），则商家可以得到来自这个用户收益的更多的分成，目前是85%。这个订阅时间不包括免费试用期。 同时可以有60天的宽限。也就是说，这一年中，如果用户曾经停止续费，然后又开始继续续费，只要中间不续费的时间不超过60天就行。