飞天诚信

OTP Server 手机令牌入门指南

V3.0

飞天诚信科技股份有限公司

网址：www.FTsafe.com

章节目录

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

第1章 概述

1.1 关于文档

本文档主要是对手机令牌进行整体介绍，包括令牌的使用以及与相关其他系统（例如 OTP Server4.8 身份认证系统）之间的关联关系。

本文在介绍手机令牌与其他系统关系时以 OTP Server4.8 身份认证系统为例，在介绍与令牌相关的功能时以 C200 类型令牌为例。

1.2 相关文档

《OTP Server 用户门户用户手册 V4.8》

《OTP Server 管理中心用户手册 V4.8》

《OTP Server 管理员指南 V4.8》

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

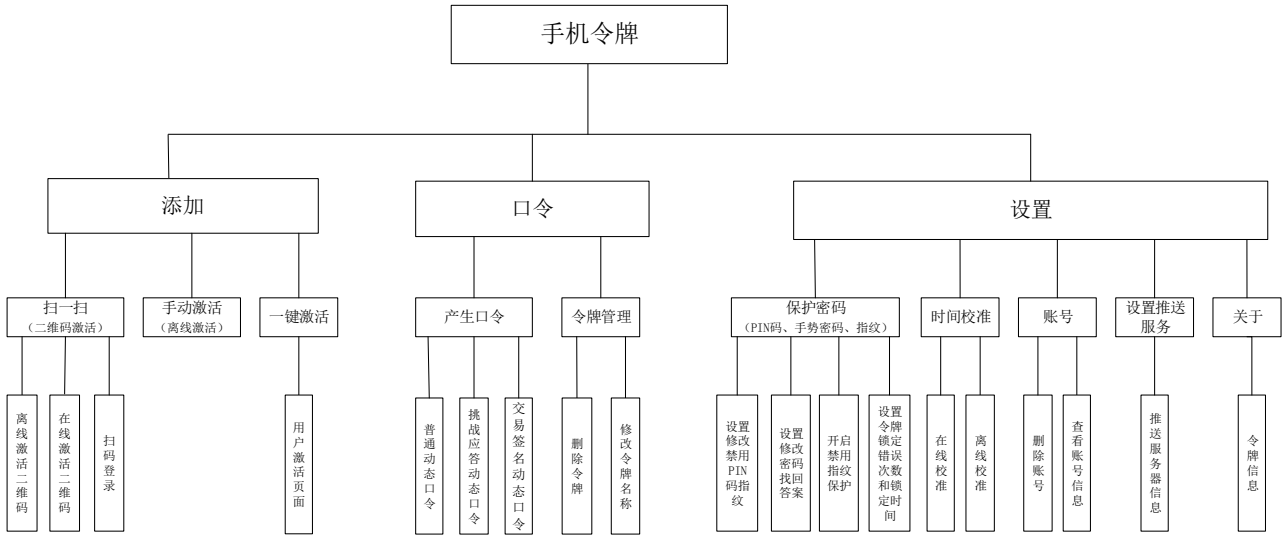
第2章 手机令牌介绍

2.1 手机令牌简介

手机令牌的主要功能是激活由认证系统分发的令牌并生成动态口令，由一个手机令牌应用程序和一个或者多个手机令牌组成，支持普通动态口令、挑战应答、交易签名等多种认证方式，可以帮助用户提高身份认证的安全性，保护用户的账户安全。将手机令牌与身份认证系统（例如 OTP Server 4.8 身份认证系统）相结合，可以为用户提供一个使用简单、安全性高、稳定性好、成本低廉的双因子身份认证解决方案。

手机令牌有三大块功能：添加、口令、设置。

总体功能框架图如下：



图表 1 手机令牌功能结构图

2.2 手机令牌适用环境

本文档所介绍的是标准版手机令牌 3.0（下面简称手机令牌），可在目前流行的 Android 和 iOS 手机平台上运行，具体所支持的版本如下：

Android 手机平台：Android4.0 及以上版本，需要至少 12M 空间；

iOS 手机平台：iOS8 及以上版本，需要至少 12M 空间；

支持的OTP Server版本：3.0及以上版本；

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

2.3 手机令牌特性

1、安全性高

2、支持的令牌类型丰富

3、应用广泛

4、易于部署

5、使用方便

6、经济实惠

7、可定制化服务

版权所有©飞天诚信科技股份有限公司

2.4 手机令牌接入方式

手机令牌有与认证系统相结合和接口两种方式：

1）与认证系统相结合方式，是指使用我们的3.0及以上版本的OTP Server身份认证系统作为后台系统；

2）调用接口方式，是指不使用我们的后台系统直接调用我们的接口来使用手机令牌。

2.4.1 与认证系统相结合方式

使用与 OTP Server 认证系统相结合的接入方式是以 OTP Server 认证系统作为手机令牌分发和认证的后台系统，手机令牌激活其分发的令牌种子后，生成动态口令再发送到认证系统进行认证。

2.4.2 调用接口方式

2.4.2.1 接口应用

1、令牌认证

进行非会话进行动态口令身份认证可以使用下面的方法。其中 auth 方法根据用户名进行认证，authToken 方法直接根据令牌号进行认证。在方法内部实现通讯初始化和反初始化。

```
auth();
authToken();
```

将通讯初始化和动态口令身份认证分开可以调用下面的方法，其中 sessionInit 方法用于创建客户端与服务器之间的会话连接，sessionVerifyOtp 方法用于进行动态口令认证。可以只进行一次通讯端口初始化，进行无数次动态口令认证。

```
sessionInit();
sessionVerifyOtp();
sessionUnInit();
```

2、令牌同步

令牌同步是为了解决令牌内计算动态口令的动态因子与认证服务器中验证动态口令的动态因子偏差过大而引入的一种机制。

进行非会话动态口令同步可以使用下面的方法。其中 sync 方法根据用户名进行同步，syncToken 方法直接根据令牌号进行同步。在方法内部实现通讯初始化和反初始化。

```
sync();
syncToken();
```

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

将通讯初始化和动态口令同步分开可以调用下面的方法，其中 sessionInit 方法用于创建客户端与服务器之间的会话连接，sessionSyncOTP 方法用于进行同步。可以只进行一次通讯端口初始化，进行无数次动态口令同步。

```
sessionInit();
sessionSyncOTP();
sessionUnInit();
```

3、 扫码登录

扫码登录是为了实现快速便捷的登录。

进行扫码登录可以使用下面的方法。其中 getAssertionId()方法是获取扫码登录的二维码信息，而方法 getAssertion()是用于判断扫码登录是否成功。

```
getAssertionId();
getAssertion();
```

2.4.2.2 开发方法

1、auth();

语法

```
int auth(String userId, String password);
```

参数

userId，表示要进行身份认证的用户名称。

password，表示用户认证时的静态密码、动态口令，如果需要第三方认证，还包括第三方认证密码；可以为上述三种类型的密码中的一个(只验证静态密码、只验证 OTP、或只验证第三方认证密码)或多个，多个密码的顺序为：第三方认证密码 + 静态密码 + 动态口令，依次输入，中间没有分割符号。

返回值

该函数的返回值为一个整数，如果返回值为 0，表示用户认证成功，如果返回值为 13，表示用户需要同步，如果返回值为其它值，表示认证失败，具体原因根据返回值对照错误代码表确定。

注释

本函数用于用户身份认证，进行身份认证时，根据用户提供的用户名、静态密码和动态口令判断用户是否为合法用户，当该用户的认证策略设置为需要静态密码时，会要求用户提供静态密码并且在认证时会检查提供的静态密码和认证服务器上所保存的静态密码是否一致，只有两者一致时，静态密码认证才能通过。

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

当用户的认证策略为不需要静态密码时，进行身份认证时将忽略静态密码检查，只要用户提供的动态口令验证通过，用户的身份认证即可通过。

举例

下面是通过 auth 函数进行用户身份认证的代码。

```
import ft.otp.agent.*;

OTPAgent agent = new OTPAgent();
String strUser = "test";
String password = "123456";
String otppassword= "";

String stracf = "C:\\windows\\system32\\otpagent.acf";
agent.setConfig(stracf);

System.out.format("Please input your otppassword to auth: ");
Scanner scanner = new Scanner(System.in);
otppassword = scanner.nextLine();

password = password + otppassword;

int strReturnCode = agent.auth(strUser, password);

if (strReturnCode == 0)
System.out.format (" authentication OK!\n");
else if (strReturnCode == 13 )
System.out.format (" Need Sync.\n");
else
System.out.format (" auth () failed with " + strReturnCode);
```

2、authToken();

authToken 方法用于通过令牌号验证用户的身份，与 auth 方法所不同的是，本方法根据令牌序列号直接进行认证，而不需要根据用户名去查找绑定的令牌，所以该方法可以用于没有与用户绑定的令牌进行动态口令认证。

语法

```
int authToken(String tokenSN, String otp);
```

参数

tokenSn，用于标识令牌的序列号字符串。

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

otp，用于标识令牌的动态口令。

返回值

该函数的返回值为一个整数，如果返回值为 0，表示用户认证成功，如果返回值为其它，表示认证失败，具体原因根据返回值查询错误代码表。

注释

本函数用于对令牌进行身份认证，由于不涉及与令牌绑定的用户，所以可以对未绑定的令牌进行认证，具有更大的灵活性。

举例

下面是通过 authToken 函数进行用户身份认证的代码。

```
import ft.otp.agent.*;

OTPAgent agent = new OTPAgent();
String strTokenSn = "3009000000006";
String strOTP = "995851";

String strAcf = "C:\\windows\\system32\\otpagent.acf";
agent.setConfig(strAcf);

int strReturnCode = agent.authToken(strTokenSn,strOTP);

if (strReturnCode == 0)
System.out.format(" authentication OK!\n");
else
System.out.format("authToken() failed with " + strReturnCode);
```

3、sessionInit();

sessionInit 方法用于初始化一个与认证服务器之间的会话，通过这个会话可以完成各种功能，比如身份认证、设置静态密码、令牌同步等。

语法

```
int sessionInit();
```

参数

该函数没有参数。

返回值

如果返回值为 0，表示初始化会话成功，如果返回值是其它值，表示初始化会话失败，具体原因可以查询错误代码表。

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

注释

会话是指认证代理和认证服务器之间的通讯过程，初始化会话就是对认证代理和认证服务器之间进行通讯以前所必需的准备工作。

举例

下面是通过 sessionInit 方法初始化会话的示例代码。

```
import ft.otp.agent.*;

OCRAAgent agent = new OCRAAgent();

String strAcf = "C:\\windows\\system32\\otpagent.acf";

int strRet = agent.init(strAcf);
if (strRet == 0)
{
strRet = agent.sessionInit();
if (strRet == 0)
{
System.out.format(" Session Init OK!\n");
agent.sessionUnInit();
}
else
{
System.out.format(" sessionInit() failed with " + strRet);
}
agent.unInit();
}
else
{
System.out.format(" init() failed with " + strRet);
}
```

4、sessionVerifyOtp();

sessionVerifyOtp 方法用于验证用户身份，其中的动态口令只能是 OTP c100/200/400 令牌或 OTP c300 令牌生成的时间动态口令，而不能用于 OTP c300 令牌生成的挑战应答动态口令验证，对挑战应答动态口令的验证应该使用 sessionAuth 方法。

语法

```
int sessionVerifyOtp(String strOTP);
```

参数

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

strOTP，用于执行身份验证的动态口令。

返回值

该函数的返回值为一个整数，如果返回值为 0，表示用户身份验证成功，否则，表示用户身份验证失败。

注释

本函数用于实现基于动态口令的身份验证。需要注意的是，使用该方法只能对事件同步或时间同步的动态口令进行认证，而不能进行挑战应答动态口令的验证。

举例

下面是通过 sessionVerifyOtp 函数进行身份验证示例代码。

```
import ft.otp.agent.*;

OCRAAgent agent = new OCRAAgent();

String strAcf = "C:\\windows\\system32\\otpagent.acf";

int strRet = agent.init(strAcf);
if (strRet == 0)
{
strRet = agent.sessionInit();
if (strRet == 0)
{
String strUser = "test";
strRet = agent.sessionSetUserName(strUser); //设置会话用户名(如果需要时设置)
if (strRet == 0)
{
System.out.format(" Session set username OK!\n");
}
else
{
System.out.format(" sessionSetUserName() failed with " + strRet);
}

agent.sessionSetTokenSN(tokenSN); //设置会话令牌号(如果需要时设置，但二者至少需要设置其中一个)

System.out.format(" Please input otp:");
Scanner scanner = new Scanner(System.in);
String strOTP = scanner.nextLine();

strRet = agent.sessionVerifyOtp(strOTP);
if (strRet == 0)
```

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

```
{
System.out.format(" log on OK!\n");
}
else
{
System.out.format(" sessionVerifyOtp() failed with " + strRet + "\n");
}
agent.sessionUnInit();
}
else
{
System.out.format(" sessionInit() failed with " + strRet);
}
agent.unInit();
}
else
{
System.out.format(" init() failed with " + strRet);
}
```

5、sessionUnInit();

sessionUnInit 方法用于结束认证代理与认证服务器之间的会话。

语法

```
void sessionUnInit();
```

参数

该函数没有参数。

返回值

该函数没有返回值。

注释

本函数用于结束一个已经完成任务的会话。

举例

下面是通过 sessionUnInit 函数结束会话的示例代码。

```
import ft.otp.agent.*;

OCRAAgent agent = new OCRAAgent();

String strAcf = "C:\\windows\\system32\\otpagent.acf";
```

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

```
int strRet = agent.init(strAcf);
if (strRet == 0)
{
strRet = agent.sessionInit();
if (strRet == 0)
{
System.out.format(" Session Init OK!\n");
agent.sessionUnInit();
}
else
{
System.out.format(" sessionInit() failed with " + strRet);
}
agent.unInit();
}
else
{
System.out.format(" init() failed with " + strRet);
}
```

6、 getAssertionId();

getAssertionId 方法用于获取扫码登录的二维码信息。

语法

```
void getAssertionId();
```

参数

该函数没有参数。

返回值

该函数的返回值为一个 OTPResult，如果其中 getResultCode()对应的值为 0，表示获取登录唯一标识成功，此时可以通过 getAssertionId ()来读取具体的值，否则，表示获取登录唯一标识信息失败。

注释

本函数用于获取扫码登录的二维码信息。

举例

下面是通过 getAssertionId 函数获取二维码信息的示例代码。

```
import ft.otp.agent.*;
```

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

```
OCRAAgent agent = new OCRAAgent();

String strAcf = "C:\\windows\\system32\\otpagent.acf";

int strRet = agent.init(strAcf);
if (strRet == 0)
{
OTPResult otpResult = agent.getAssertionId();
int resultCode = otpResult.getResultCode();
   if(resultCode == 0) {
       System.out.format("getAssertionId is Success: ", resultCode);
       System.out.format("AssertionId is : ", otpResult.getAssertionId());
   } else {
       System.out.format("getAssertionId failed, error code is :", resultCode);
  }
else
{
System.out.format(" init() failed with " + strRet);
}
```

7、getAssertion();

getAssertion 方法用于判断扫码登录成功与否。

语法

```
void getAssertion();
```

参数

assertionId:二维码信息。

返回值

该函数的返回值为一个整数，如果返回值为 0，表示用户扫码登录成功，否则，表示用户扫码登录失败。

注释

本函数用于判断用户扫码登录成功与否。

举例

下面是通过 getAssertion 函数的示例代码。

```
import ft.otp.agent.*;

OCRAAgent agent = new OCRAAgent();

String strAcf = "C:\\windows\\system32\\otpagent.acf";
```

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

```
int strRet = agent.init(strAcf);
if (strRet == 0)
{
OTPResult otpResult = agent.getAssertion();
int resultCode = otpResult.getResultCode();
   if(resultCode == 0) {
       System.out.format("getAssertion is Success: ", resultCode);
   } else {
       System.out.format("getAssertion is failed, error code is :", resultCode);
   }
else
{
System.out.format(" init() failed with " + strRet);
}
```

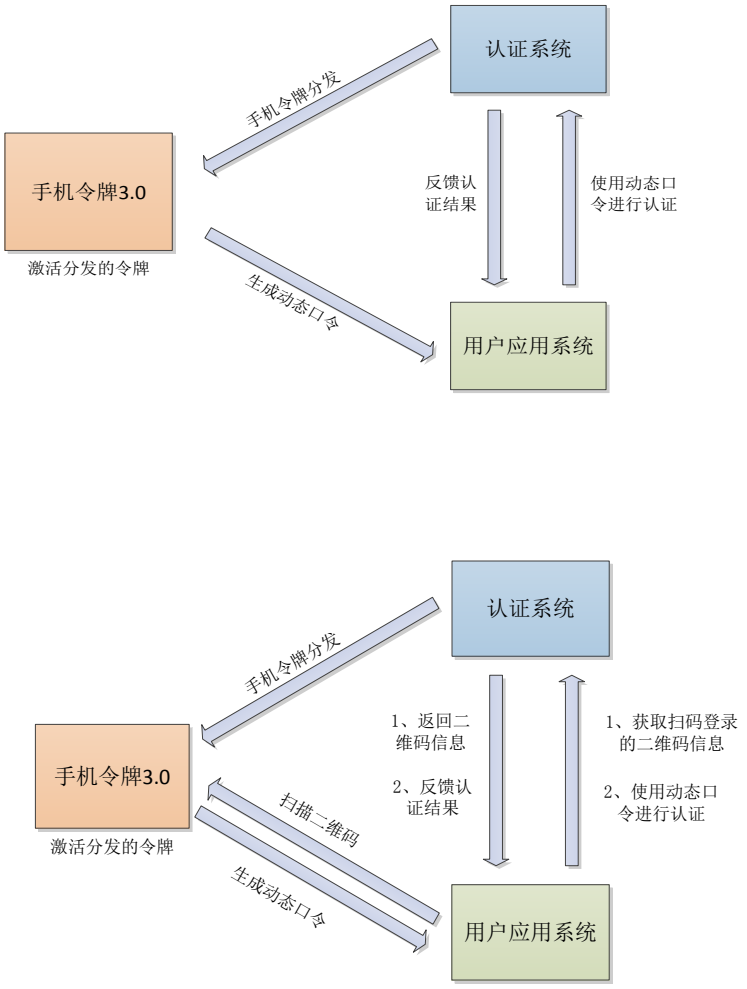
版权所有©飞天诚信科技股份有限公司

2.5 手机令牌与其他系统关系

手机令牌在使用时与认证系统、用户应用系统（例如网银客户端系统）相结合。

通过认证系统将手机令牌分发后，可使用手机令牌 App 对分发的令牌进行激活并生成动态口令。用户在应用系统中，使用产生的动态口令进行身份认证操作。例如用户在网上银行进行转账操作时，网银客户端系统要求用户输入正确的动态口令后，才允许继续操作。

具体交互过程如下图所示：

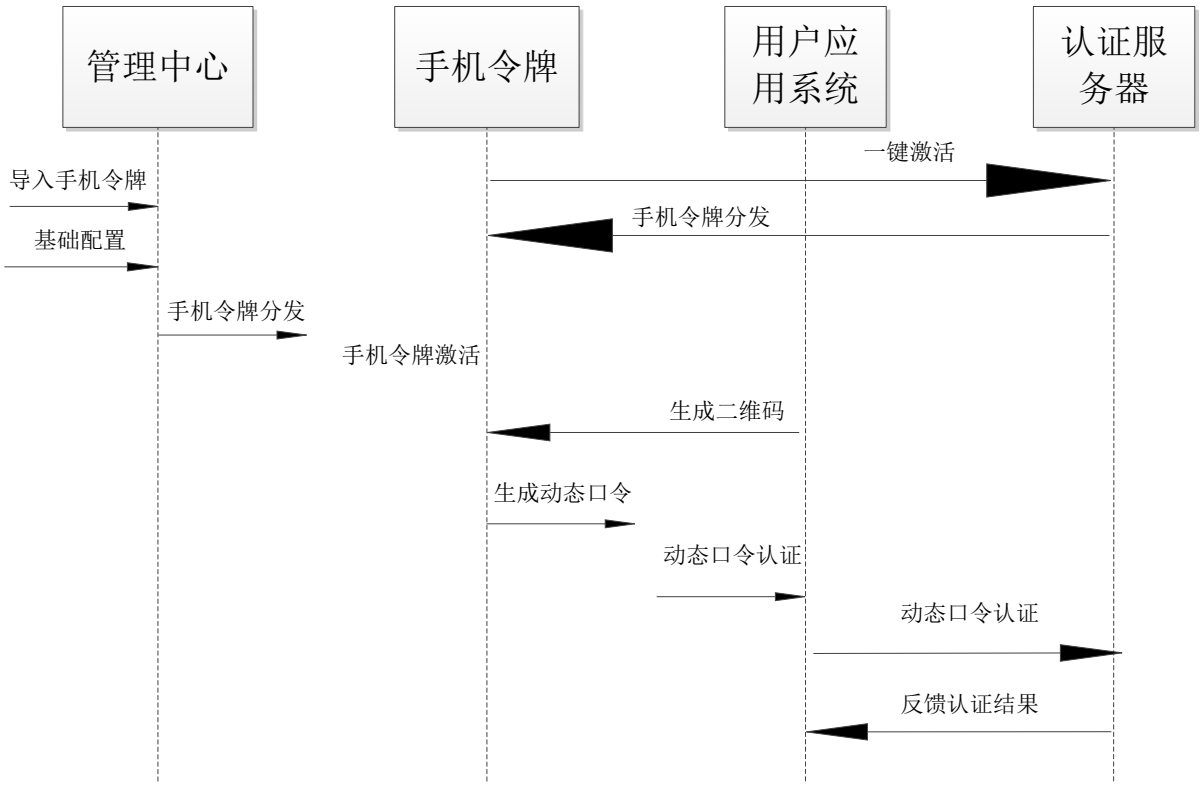


图表 2 手机令牌与其他系统交互图

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

第3章 手机令牌使用

实际应用中手机令牌应用程序的使用与认证系统相结合，下面以 OTP Server4.8 身份认证系统为例来说明手机令牌的实际应用，整体操作流程如下图所示：



图表 3 手机令牌实际应用流程图

下面具体介绍操作方法：

3.1.1 管理中心

管理中心的主要功能是对系统参数进行配置、对所有令牌进行管理、对令牌进行分发、认证等。针对手机令牌，主要是做一些基础配置和将令牌分发给用户的操作。

3.1.1.1 手机令牌导入

导入手机令牌操作在管理中心的“令牌管理”→“令牌导入”中进行，选择相应的组织机构、文件格式和令牌文件进行导入，“是否启用令牌”是指将令牌导入管理中心后是否设置令牌状态为启用，启用状态下的令牌可以进行正常使用。

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

图表 4 令牌导入

3.1.1.2 基础配置

1、 为了确保对手机令牌分发的正确性，对手机令牌分发前，需要对“激活密码策略”和“分发站点策略”进行设置。

2、 手机令牌一键激活功能，需要开启“手机令牌推送策略”。

图表 5 手机令牌配置界面

具体配置项如下：

激活密码策略

激活密码有效周期（天）：在分发手机令牌时若选择需要激活密码，则在生成手机令牌激活信息时还会生成激活密码，用户在激活令牌时需要输入相应的激活密码才能成功激活令牌。

该项设置用于限制激活密码的有效期，在该设置时间内，激活密码可用，超过设置的时间之后激活密码将失效。

在线分发激活密码重试次数：该字段设置手机令牌激活密码重试次数，该设置在手机程序端激活在线分发的二维码图片时起作用，如果输入错误次数超过设置的值就不允许再进行激活，必须重新进行分发。

默认激活密码：在手机令牌分发中，激活密码选择“默认”时，显示的激活密码为此处设置的激活密码。系统默认为 123456，可进行修改。

启用短信发送离线分发信息：选“是”，手机令牌离线分发时，将通过短信的形式，将分发信息发送到，令牌绑定的用户填写的手机号码上。选“否”，手机令牌分发时，不给绑定的用户手机发送分发信息。

启用邮件发送分发信息：选择“是”，当手机令牌绑定了用户且设置了邮箱时，手机令牌分发信息，将以邮件形式通知该用户。选择“否”，手机令牌分发时，不发送邮件给其绑定的用户。

提示：只适用于手机令牌被一个用户绑定的情况。

分发站点策略

分发站点访问类型：通过下拉选项选择分发站点访问的类型：http、https、http 和 https。

分发站点访问 URL：填写分发站点访问的 URL，分发站点访问 URL 中的 IP 地址需输入 web 服务所在机器的 IP 地址，用于在线激活手机令牌时的认证功能，在线激活只能进行一次，激活之后的二维码不能再激活。

是否启用分发站点：选择“启用”，则启用分发站点，手机令牌在线分发时可以生成二维码；选择“关闭”，则停用分发站点，手机令牌在线分发时不能生成二维码。

提示：分发站点策略配置主要用于在线分发二维码，所配置的信息会体现在分发二维码图片中，若要保证手机令牌能成功的在线分发，需确保分发站点信息配置正确并且处于启动状态。

手机令牌一键激活

是否启用手机令牌推送：选择“启用”，则启用手机令牌推送策略，页面显示配置推送服务器相关信息；选择“关闭”，则手机令牌不能推送登录，页面不显示配置推送服务器相关信息。

提示：启用手机令牌推送策略，主要体现在“令牌管理→手机令牌分发→二维码”功能模块。

推送服务器名称：必填项，输入 1-64 个字符以内的推送服务器名称，默认为 publishServer，也可以改

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

为其它指定的服务器名称。

推送服务器 IP：必填项，输入推送服务器的 IP 地址。

提示：推送服务器 IP 地址，页面显示的值是认证服务器所在系统环境的真实 IP 地址。

推送服务器端口：必填项，输入推送服务器的端口，默认为 58004，也可以改为其它指定的端口，需要修改“pushserver.properties”文件里面的推送服务器端口（文件路径：/otpserver/otpauthservice/conf/），重启服务器生效。

“保存”按钮：选择“启用”，推送服务器配置信息填写正确，点击“保存”按钮，手机令牌推送策略配置成功。

提示：

1、开启手机令牌推送策略，需要修改“pushserver.properties”文件里面的认证服务器 IP，文件路径：

/otpserver/otpauthservice/conf/

2、相关文件配置完成，管理中心开启了手机推送策略后，后台需要开启推送服务器，Linux 环境下执行：

./ startpushserver.sh，Windows 环境直接双击运行：start_otp_push_server.bat，运行推动服务器脚本路径：

/otpserver/otpauthservice/bin/

一键激活和一键解绑认证方式：默认选择的是“后端认证”，在手机端操作手机令牌一键激活功能，管理中心需要配置后端服务器（认证管理→后端认证→添加后端认证），用户需要输入后端服务器密码完成一键激活；“动态口令认证”：用户已绑定令牌，在手机端操作手机令牌一键激活功能时，用户需要输入已绑定令牌的口令完成一键激活；“静态密码认证”：在手机端操作手机令牌一键激活功能时，用户需要输入用户的静态密码完成一键激活；“自定义接口认证”：OTPServer 服务器对接其它认证系统，此功能需要定制开发。

认证成功自动创建不存在用户：默认选择“创建”，操作手机端的手机令牌进行一键激活，如果该用户在本地不存在，则会自动创建。选择“不创建”，如果本地不存在该用户，操作手机端的手机令牌进行一键激活，则一键激活失败；如果本地用户存在，操作手机端的手机令牌进行一键激活，则一键激活成功。

提示：此项功能与“一键激活和一键解绑认证方式”结合使用，上述项选择是“后端认证”、“自定义接口认证”，界面自动显示此项。

解绑一键激活的手机令牌：该选项与手机令牌一键激活功能有关，选择“启用”，在手机端删除一键激活的令牌时同时解绑该令牌与用户的绑定关系；选择“关闭”，只是把该令牌从手机端删除了，并没有解除与用户的绑定关系。

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

手机令牌一键激活二维码：在手机端操作手机令牌进行一键激活时需要设置推送服务器信息，可直接扫该二维码进行设置推送服务器信息。

3.1.1.3 令牌分发

管理中心中的手机令牌，需要经过分发生成二维码图片或激活信息，才能被手机令牌程序应用激活。

令牌分发功能在管理中心的“令牌管理”→“手机令牌分发”模块中，有“二维码”和“离线分发”两种分发方式。

二维码分发，最终产生一个可扫描的二维码图片。离线分发，最终会产生一组分发信息，包括令牌号、激活码和激活密码，在手机令牌中手动输入激活信息激活。

图表 6 二维码分发手机令牌

图表 7 离线分发手机令牌

其中“二维码”分发有在线和离线两种方式，在线分发的二维码图片只能被激活一次，安全性更高。

3.1.2 手机令牌

应用分为三个功能页面：添加、口令、设置，添加界面可以通过“扫一扫”、“手动激活”和“一键激

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

活”来激活分发的令牌；口令界面是使用已经激活的令牌生成动态口令的；设置界面可以对令牌程序进行参数设置。

3.1.2.1 安装手机令牌程序

Android 手机需要从各大应用市场去下载（ios 手机需要从 AppStore 中下载）并安装手机令牌应用，安装界面显示安装导航，可根据导航进行该操作。

提示：应用市场包括华为应用市场、腾讯应用宝、oppo 应用商店、360 手机助手、豌豆荚（PP 助手）、搜狗手机助手、安智市场、google play

3.1.2.2 激活手机令牌

管理中心分发的令牌种子信息可以通过邮件或者短信发送给用户，手机程序通过分发信息进行令牌激活。

图表 8 手机令牌添加界面

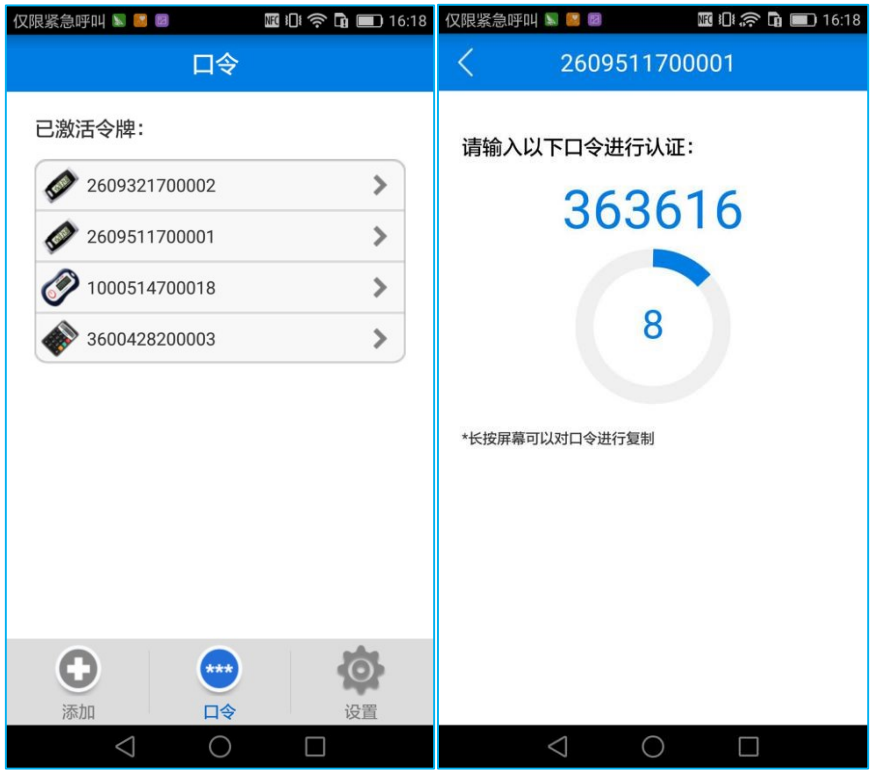
如果分发时选择的是“二维码”方式，则用手机令牌的“扫一扫”功能激活令牌。如果分发时选择的是“离线分发”方式，则用手机令牌的“手动激活”功能激活令牌，令牌激活后会显示在“口令”界面。如果选择“一键激活”功能激活令牌，则输入用户名和密码进行激活，令牌激活后会显示在“口令”界面，并且用户信息在“设置”界面中的“账号管理”界面显示。

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

提示：使用“一键激活”功能激活令牌，首先要配置推送服务器信息，并保证管理中心中推送服务器配置开启以及推送服务器正常运行。

3.1.2.3 生成动态口令

当用户使用手机令牌登录应用系统或进行网上交易时，需要进行动态口令认证，打开手机令牌应用程序，在“口令”界面，点击对应的手机令牌跳转到口令界面。



图表 9 手机令牌口令界面

3.1.3 用户门户

用户门户是一个方便用户对自己的令牌进行一些基础管理，像申请、解锁、挂失、认证、同步等操作的平台，它是从管理中心中开放出来的部分功能，与管理中心连接同一个数据库，用户所能查看和操作的数据信息也是从管理中心中开放出来的。

针对手机令牌，用户可以进行令牌分发、令牌认证、同步等操作证操作，实际操作时需要与手机令牌程序配合进行。

3.1.3.1 令牌分发

若用户已经绑定了手机令牌，则选择要分发的手机令牌进入分发界面，用户门户中的令牌分发功能与管理中心中的手机令牌分发功能相同，可参照 3.1.1.3 内容。

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

若用户还未绑定手机令牌，则需要先申请绑定，系统会自动从对应组织机构下所有未绑定的手机令牌中随机给用户绑定一个，然后再进行令牌分发。

图表 10 申请绑定手机令牌

图表 11 手机令牌分发

3.1.3.2 令牌认证

令牌认证测试是用来给用户检查所绑定的令牌能否成功使用的，在手机令牌程序的“口令”界面找到，对应的手机令牌种子生成动态口令，输入认证窗口中认证，口令正确时认证成功。

提示：在实际业务中，用户在使用手机令牌保护的应用系统时，也同样需要相应的手机令牌生成动态口令进行身份认证，输入的动态口令正确才能继续使用应用系统。

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com

图表目录

版权所有©飞天诚信科技股份有限公司
公司网址： www.FTsafe.com