ETz201 技术分析

OTP 简介：

OTP 全称叫 One-time Password,也称动态口令，是根据专门的算法每隔 60 秒生成一个与时间相关的、不可预测的随机数字组合，每个口令只能使用一次，每天可以产生 1440 个密码。

用户进行认证时候，除输入账号和静态密码之外，必须要求输入动态密码，只有通过系统验证，才可以正常登录或者交易，从而有效保证用户身份的合法性和唯一性。动态口令最大的优点在于，用户每次使用的口令都不相同，使得不法分子无法仿冒合法用户的身份。

动态口令认证技术被认为是目前能够最有效解决用户的身份认证方式之一，可以有效防范黑客木马盗窃用户账户口令、假网站等多种网络问题，导致用户的财产或者资料的损失。

相比较动态口令认证方式，静态口令认证缺点如下：

（1） 为了便于记忆，用户多选择有特征作为密码，所有静态口令相比动态口令而言，容易被猜测和破解；

（2） 黑客可以从网上或电话线上截获静态密码，如果是非加密方式传输，用户认证信息可被轻易获取；

（3） 内部工作人员可通过合法授权取得用户密码而非法使用；

静态口令根本上不能确定用户的身份，其结果是，个人可以轻松地伪造一个假身份或者盗用一个已有使用者的身份，给企业造成巨大的经济和声誉损失。

针对这种情况，目前已知的有 ET 系列基于”时间”（Time）形式，从安全性、可靠性、稳定性、可扩展性的做的相当好，并且 ET 系列 OTP 认证服务器可以很好的支持负载均衡，单台计算机可以支持高 10000 客户访问量，已经应用到很多身份认证应用当中。

ETz201 认证方式：

ET z201 基于“时间”（Time）形式。每个动态口令的产生是硬件中 160 位密钥与时钟芯片内的时间计算产生，每隔 60 秒变化一次。ET z201 采用符合 OATH 国际标准组织的 TOTP 算法，安全可靠。

ET z201 同步操作：

当 OTP 令牌中的时钟与 OTP 认证服务器上的时钟误差范围过大，超过认证范围时，就需要进行同步操作。

可以修改同步范围，默认的是 40，表示为 OTP 认证服务器时间的前后各 20 分钟范围内。如果同步失败，可以调大这个范围。

详细说明：如硬件中的时间是 8:00，而服务器（调用接口的计算机）的标准时间为 9:00，接口中 authwnd 参数默认为 20，即 9:00 的前后各 10 分钟，那么硬件中时间的范围在 8:50-9:10 时，硬件产生的 OTP 是可以认证通过的，但硬件中为 8:00，验证就不能通过。这时可以通过 2 个办法来解决这个问题，一种是调整 authwnd 参数，设为 120，那么就是 9:00 的前后各 1 小时范围内，即 8:00-10:00，硬件中为 8:00，在这个范围，可以验证通过。这种方法增加了服务器运算的范围，加重了负荷，因此不建议使用。另外一个解决办法就是同步运算，同步窗口参数 syncwnd 可以设成 120，同步成功后将调整值 drift 和 succ 存入到数据库，供认证接口调用。认证接口再次调用时传入的是同步成功后的调整值，就能够弥补动态令牌硬件中时间和服务器时间的偏差了。由于同步接口只是在时间偏差过大，造成认证不通过才调用，被调用的概率很小，不会增加服务器计算的负荷，建议使用这种方法。

ETz201 接口定义如下：贴出来与大家分享

```
*
================================================================================
 * Function     : ET_CheckPwdz201
 * Description: OTP Z201(TOTP)  认证接口
 * Parameter    :
 *          authkey             令牌密钥
 *          t                   当前时间相对UTC Epoch秒数
 *          t0                  起始参考时间相对UTC Epoch秒数(默认为)
 *          x                   TOTP变化周期(默认为秒)
 *          drift               漂移次数
 *          authwnd             认证范围, 通常是-20
 *          lastsucc            前一次认证成功的相对UTC Epoch秒数(为防止重放攻击)
 *          otp                 需要认证的动态口令
 *          otplen              需要认证的动态口令长度, 通常是
 *          currsucc        认证成功后的相对UTC Epoch秒数
 *          currdft             认证成功后的当前漂移次数
 *
 * return       : 0 -  成功, 其他值为错误.
 */
int __stdcall ET_CheckPwdz201(char *authkey, uint64_t t, uint64_t t0,
      unsigned int x, int drift, int authwnd, uint64_t lastsucc,
      const char *otp, int otplen, uint64_t *currsucc, int *currdft);

/*
================================================================================
 * Function     : ET_Syncz201
 * Description: OTP Z201(TOTP)  同步接口
 * Parameter    :
 *          authkey             令牌密钥，已经加密过的，需要对其进行解密
 *          t                   当前时间相对UTC Epoch秒数
 *          t0                  起始参考时间相对UTC Epoch秒数(默认为)
 *          x                   TOTP变化周期(默认为秒)
 *          drift               漂移次数
 *          syncwnd             同步范围, 通常是-20
 *          lastsucc            前一次认证成功的相对UTC Epoch秒数(为防止重放攻击)
 *          otp1                需要同步的第一个动态口令
 *          otp1len             需要同步的第一个动态口令长度, 通常是
 *          otp2                需要同步的第二个动态口令
 *          otp2len             需要同步的第二个动态口令长度, 通常是
 *          currsucc        认证成功后的相对UTC Epoch秒数
 *          currdft             认证成功后的当前漂移次数
 *
 * return       : 0 -  成功, 其他值为错误.
 */
```

```
int __stdcall ET_Syncz201(char *authkey, uint64_t t, uint64_t t0,
        unsigned int x, int drift, int syncwnd, uint64_t lastsucc,
        const char *otp1, int otp1len, const char *otp2, int otp2len,
        uint64_t *currsucc, int *currdft);
```

服务端开发认证方法

（1） 在您的数据库中增加一张用于存储OTP动态令牌信息的数据表。里面至少存储以下字段：“令牌号”（背面条形码）、“密钥”（authkey）、“成功值”（currsucc）、“漂移值”（currdft）。其中令牌号和密钥都可以用字符串形式，成功值和漂移值接口中要用到uint64和int类型。

（2） 在系统的用户表中增加一个存“令牌号”的字段，存储与用户绑定的令牌号。

用户在登录时，输入用户名和 OTP 传给服务器端。服务器通过用户名到用户表中得到“令牌号”，再通过这个“令牌号”到令牌表中得到“密钥”，“成功值”和“漂移值”，带入到接口中进行认证或同步，认证或同步成功后将返回的值写回数据库中保存。认证或同步失败时，不要将这两个值写回数据库。

使用 ETz201 进行服务端的认证 ，就是这么简单。

ET z201 优势:

国内唯一一家国际 OATH 组织成员

算法为国际标准 HOTP/TOTP 算法

OTP 系统支持负载均衡，即 OTP 服务器的集群。目前测试一台 PC 装置的 OTP 服务器可承受近 10000 个并发访问

硬件质量稳定。具有防深水浸泡，防汽车碾压，防电磁干扰等保护

管理工具采用 WEB 模式的 B/S 结构，方便管理员在不同的机器上进行管理

支持标准 Raduis 协议的各种应用

支持多种数据库：Oracle、SQL Server、 My SQL、 Access 等