北京大学
PEKING UNIVERSITY

# 动态认证系统

汇报人： 胡旭伦 张逸然 李为民 路建飞

北京大学
PEKING UNIVERSITY

# 概述

# OTP

- 全称为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。
- 分类：时间同步、事件同步、挑战/应答

- **时间同步（TOTP，Time-based One-Time Password）：**

表示基于时间戳算法的一次性密码。 即基于客户端的动态口令和动态口令验证服务器的时间进行比对，一般每60秒产生一个新口令，要求客户端和服务器能够十分精确的保持正确的时钟，客户端和服务端基于时间计算的动态口令才能一致。

- **事件同步（HOTP，HMAC-based One-Time Password）：**

表示基于HMAC算法加密的一次性密码。

- **挑战/应答：**

常用于的网上业务，在网站/应答上输入服务端下发的挑战码，动态令牌输入该挑战码，通过内置的算法上生成一个6/8位的随机数字。

# OAuth

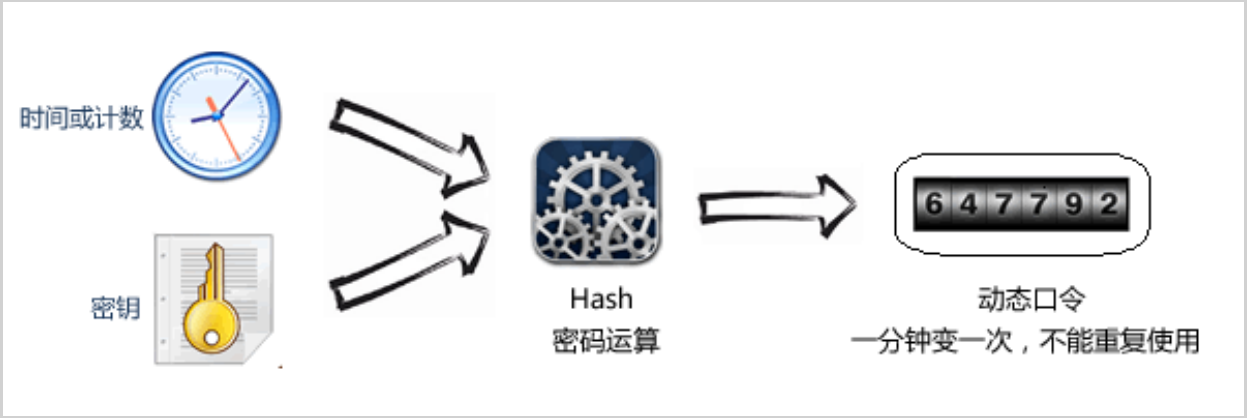
- 全称Open Authorization，即开放授权，是为用户资源的授权定义了一个安全、开放及简单的标准，第三方无需知道用户的账号及密码，就可获取到用户的授权信息，并且这是安全的。

北京大学
PEKING UNIVERSITY

# TOTP算法

TOTP算法(Time-based One-time Password algorithm)是一种从共享密钥和当前时间计算一次性密码的算法。 它已被采纳为Internet工程任务组标准RFC 6238，是Initiative for Open Authentication（OATH）的基石，并被用于许多双因素身份验证系统。

TOTP是基于散列的消息认证码（HMAC）的示例。 它使用加密哈希函数将密钥与当前时间戳组合在一起以生成一次性密码。 由于网络延迟和不同步时钟可能导致密码接收者必须尝试一系列可能的时间来进行身份验证，因此时间戳通常以30秒的间隔增加，从而减少了潜在的搜索空间。




如上图，是一种基于时间同步的OTP计算方式，是通过客户端和服务器持有相同的密钥并基于时间基数，服务端和客户端采用相同的Hash算法，计算出长度为六位的校验码。当客户端和服务端计算出的校验码相同是，那么验证通过。

由于客户端需要存储密钥和计算校验码的载体，阿里云的身份宝（或者Google 的Authenticator）提供了手机端的APP进行密钥存储和校验码计算。

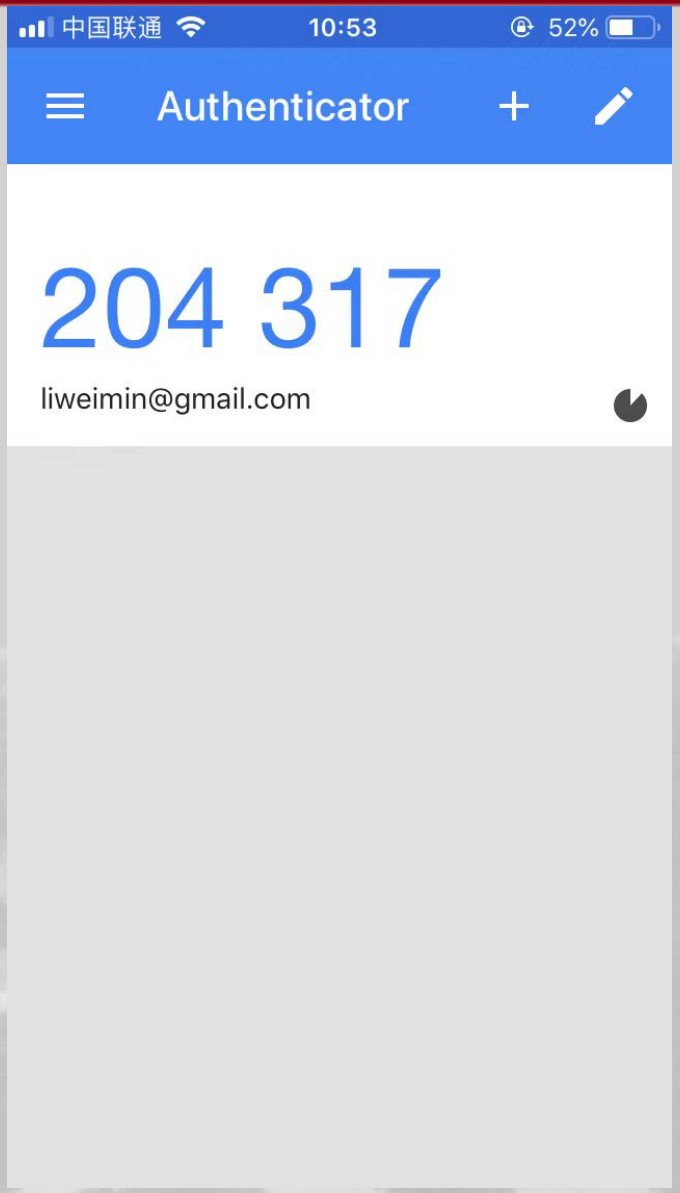
## 算法概要

TOTP（基于时间的一次性密码算法）是支持时间作为动态因素基于HMAC一次性密码算法的扩展。

本算法是一个对称算法，也就是说，后台和移动端采用同样的密钥，同时这个算法是依赖于当前的系统时间的，所以可以用于动态验证。

**TOTP = HMAC-SHA-1(K, (T - T0) / X)**

- K 共享密钥
- T 当前时间戳
- T0 开始的时间戳
- X 时间步长


中国联通
10:53
52%
Authenticator
204 317
liweimin@gmail.com

## 弱点和漏洞

TOTP代码可以像密码一样被钓鱼，但它们需要网络钓鱼者实时代理凭证，而不是及时收集它们.

不限制登录尝试的实现容易受到强制执行代码的攻击。

窃取共享密钥的攻击者可以随意生成新的有效TOTP代码。如果攻击者破坏了大型身份验证数据库，这可能是一个特殊问题。

由于TOTP设备的电池电量不足，时钟可以解除同步，并且由于软件版本在用户可能丢失或被盗的手机上，因此所有实际实施都有绕过保护的方法（例如：打印的代码，电子邮件 - 重置等），这可能给大型用户群带来相当大的支持负担，并且还为欺诈用户提供额外的利用向量。

TOTP代码的有效期超过它们在屏幕上显示的时间（通常是两倍或更多倍）。这是一个让步，认证和认证方的时钟可以大幅度扭曲。

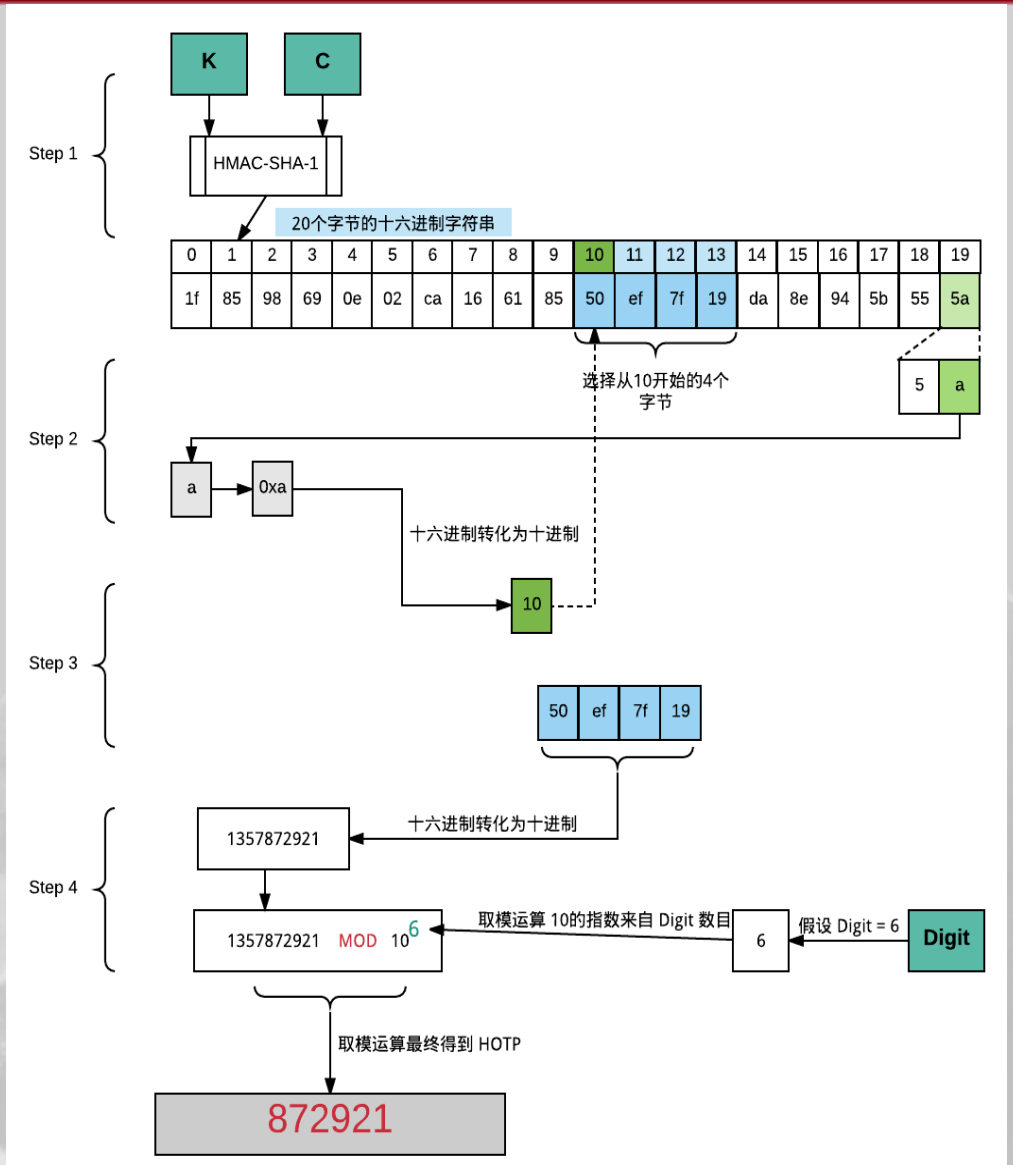
所有一次性基于密码的身份验证方案（包括TOTP和HOTP等）仍然容易受到会话劫持，即在用户登录后占用用户的会话 。

北京大学
PEKING UNIVERSITY

# HOTP算法

# 算法原理

- HOTP是一种基于哈希消息认证码（HMAC）生成一次性密码值的算法，HMAC利用哈希算法，以一个密钥和一个消息为输入，生成一个消息摘要作为输出。
- HMAC（K，C）
- K 共享密钥
- C 计数器，是一个 8个 byte的数值，而且需要服务器和客户端同步。
- Digit 表示产生的验证码的位数


K
C
Step 1
HMAC-SHA-1
20个字节的十六进制字符串
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
1f 85 98 69 0e 02 ca 16 61 85 50 ef 7f 19 da 8e 94 5b 55 5a
选择从10开始的4个字节
5 a
Step 2
a
0xa
十六进制转化为十进制
10
Step 3
50 ef 7f 19
十六进制转化为十进制
1357872921
Step 4
1357872921 MOD 10^6
取模运算 10的指数来自 Digit 数目
6
假设 Digit = 6
Digit
取模运算最终得到 HOTP
872921

# 安全性分析

- T 称为限制参数（Throttling Parameter）
- 这里当用户重试次数超过了阈值 T，服务器就应该将该账号标记为有安全风险，需要提醒用户或者拒绝该用户的连接。另外还有一个基于延时的策略还防止暴力破解攻击，服务器设置一个惩罚时间，一旦用户验证失败，需要在**惩罚时间乘以失败次数**的时间内禁止用户重新尝试验证

# 安全性分析

- s 称为重新同步参数（Resynchronization Parameter）

如果用户严格按照生成一次OTP，然后验证一次的话，服务器直接可以验证成功。因为算法将会输入相同的参数。

如果用户无意间多生成了若干次OTP但是没有用来验证，服务器和客户端计数器就产生差异，这时候服务器端会自动累加计数器来尝试匹配用户输入的 OTP，一旦服务器尝试 s 次仍未匹配成功，那么就会通知客户端需要重新生成OTP来验证，直到验证成功。

# 可改进的地方

- 1. 位数。例如对10^8取模生成8位的OTP
- 2. 使用A-Z和0-9值
- 3. 引入数据字段
- 例如HOTP（K, C, [Data]），其中Data是一个可选字段。例如，Data = Timer，其中Timer可以是UNIX时间（GMT秒自1970年1月1日起），除以某个因子（8,16,32等）以给出特定的时间步长

北京大学
PEKING UNIVERSITY

# 挑战/应答方式

基于挑战应答模式的令牌属于异步令牌，由于在令牌和服务器之间除相同的算法外没有需要进行同步的条件，故能够有效地解决令牌失步的问题，降低对应用的影响，同时极大的增加了系统的可靠性。

异步口令使用的缺点主要是在使用时，用户需多一个输入挑战值的步骤，对于操作人员，增加了复杂度。

北京大学
PEKING UNIVERSITY

# 挑战应答模式实现过程

# 挑战/应答机制工作原理

当用户试图访问一个服务器主机时，认证过程开始。服务器端在收到登录请求后即产生一个挑战信息发送给客户端，用户在客户端输入只有自己知道的通信蜜语，并由动态口令计算器产生一个OTP。此OTP通过网络送到服务器，服务器再校验此口令。若此OTP被认证成功，则客户被授权访问服务器，而此OTP将不再被使用


挑战/应答一般流程：

1. 向服务器发送请求，要求进行身份验证
2. 服务器从数据库中查找用户名是否合法，若不合法则不做处理
3. 服务器产生一个“挑战”随机数，发送给客户端，作为“提问”
4. 客户端使用“用户名”+H（以共享秘钥+挑战）做应答
5. 服务器收到应答和自己的计算做比较相等则认证通过，反之失败
6. 服务器挑战客户端成功还是失败

# 挑战/应答机制安全性分析

1. 能够有效抵抗截获/重放攻击：

即使攻击者可以通过一些软件或其他手段窃听到用户的动态口令，但是由于动态口令是一次性的，发送给认证服务器也不能通过认证。而且由于每个动态口令之间是不相关的，不能从这个动态口令推出下一个动态口令，所以攻击者得到了动态口令也没有用。

2. 中间人攻击：

挑战/应答机制没有实现双向认证，因此不能抵抗中间人攻击。由于认证协议的单向性，即服务器可以验证用户的身份而用户不能验证服务器的身份，若攻击者冒充服务器，就可以得到用户的口令，并且可以将此口令发送到合法的认证服务器。

要解决这个问题，必须采用公钥技术设计新的密码协议，但这会增加计算量，从而降低认证速度。

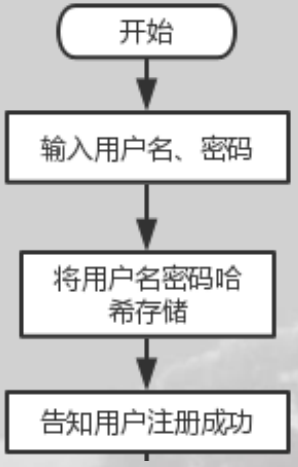
北京大学
PEKING UNIVERSITY

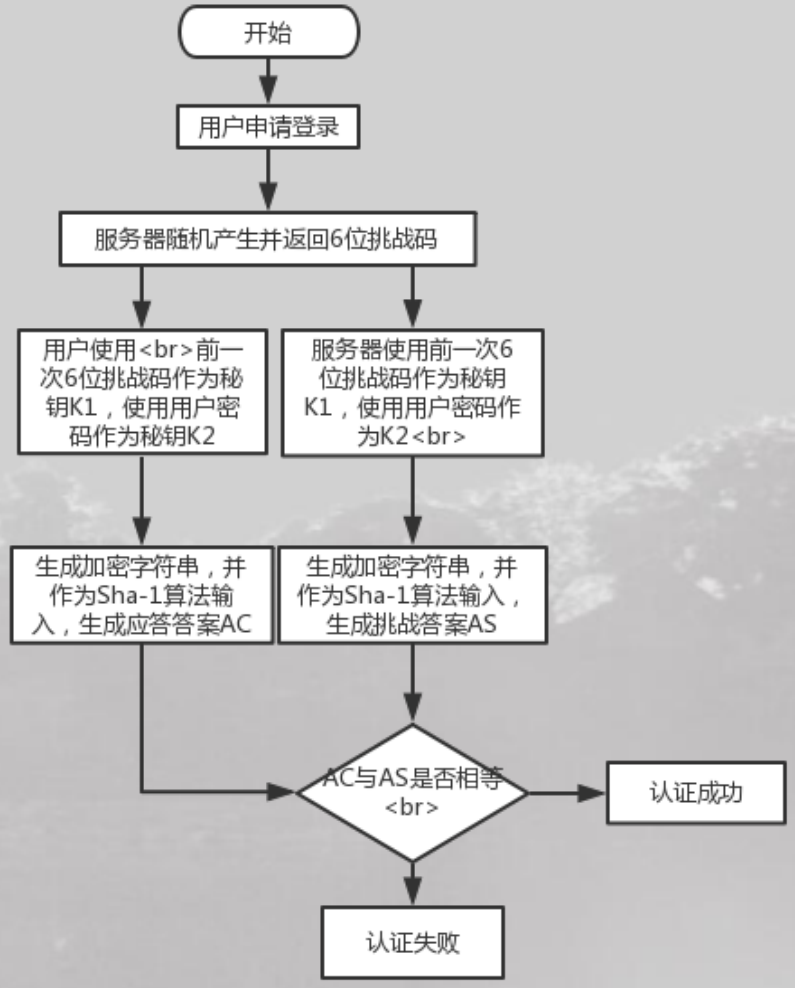
# 动态认证系统实现

# 动态认证思路

在挑战/应答机制中，双方使用前一次的挑战问题用作秘钥K1，用户密码作为秘钥K2，加密挑战问题作为应答值。

优点：1. 实现挑战验证过程的一次一密，增强安全性。
2. 需要知道前一次的挑战值和用户密码才可以解密挑战问题


开始
输入用户名、密码
将用户名密码哈希存储
告知用户注册成功


开始
用户申请登录
服务器随机产生并返回6位挑战码
用户使用<br>前一次6位挑战码作为秘钥K1，使用用户密码作为秘钥K2
服务器使用前一次6位挑战码作为秘钥K1，使用用户密码作为K2<br>
生成加密字符串，并作为Sha-1算法输入，生成应答答案AC
生成加密字符串，并作为Sha-1算法输入，生成挑战答案AS
AC与AS是否相等<br>
认证成功
认证失败

# 挑战值计算

```
For i=1 to len(strSource)
#选取待加密字符中的一个字符
    strChar = ChooseOne(strSource)
#取字符低字节和K1异或运算
    blowData = Low(strChar) XOR K1
#取字符高字节和K2异或运算
    bHigData = Hig(strChar) XOR k2
    strRes = strRes + bHigData +blowData
#散列算法
strRes = MD5(strRes)
```


北京大学
PEKING UNIVERSITY
E:\研究生学习\信息安全工程\项目\OTP1\OPT1\Debug\OPT1.exe
1. 用户注册
2. 身份认证
3. 口令计算
0. 退出
请选择:

北京大学
PEKING UNIVERSITY

# 总结

# 项目总结

- 实现了挑战/问答模式的动态口令验证
- 利用历史挑战信息实现了认证会话的一次性密钥

# 展望

- 解决用户存储历史验证数据的安全性
- 解决用户在不同客户端登录时验证服务器的问题

北京大学
PEKING UNIVERSITY

# 谢谢观看
# 望批评指正