TCST'18 - Yu-Tao Liu - SplitPass: A Mutually Distrusting Two-Party Password Manager.

优点：①从登录设备的内存可能遭受攻击的角度设计口令管理器

②利用辅助云，实现辅助云与登录设备互不信任的认证方案.

问题：①在server端组装的口令为明文，不够安全，应考虑实际中的口令认证场景

②需用户在云和从设备上分别存储份额

③重组包可能被视作IP欺骗，若允许可能导致server遭受真的IP欺骗

④应详细讨论自建的部署的难度(如应存储多少条目可平衡安全性和用户体验).

▲研究背景

1> 由于PM管理着全部的口令，是攻击者的一个目标.

①本地存储口令的PM：若攻击者可以访问登录设备，可以通过内存扫描和其它复杂的攻击获取口令

②使用辅助设备存储口令的PM：cloud可能无法被信任；使用移动设备也需要在本地对口令进行解密. ⇒需假设登录设备的内存是可信的.

另外，auto-fill的功能很危险，可能被利用

2> PM应考虑登录设备的内存不安全的情况，希望提供有辅助设备的方案.

↳ Android + public cloud.

▲研究问题

1>威胁模型.(攻击者不会同时完成①和②).

①A可以物理访问移动设备，内存和存储内容均是可访问的

②云辅助空间是不可信的，可能被攻击者破坏.

2>其它假设.

①登录设备需联网，app使用安卓提供的SSL库与服务器建立SSL连接

②假设用户已知SplitPass的存在，并且按照SplitPass要求的口令格式输入.

③SplitPass专注于保护口令不关心其它的内容，包括general device data protection

3>目标(在移动设备上提供强的口令保护)

①整个口令的明文不会在移动设备或辅助云中出现

②不需要服务器或app的更改.

→在以上的要求中，在移动设备和辅助云互不信任的情况下实现PM.用于认证.

4>解决方案概述.→共享唯一的口令.

①Android设备和云分别保存口令的一个share，在登录时共同恢复口令.双方不能获取对方的share

②两个组件建立SSL登录请求，共享SSL record encapsulation和network packet framing过程.

对server不可见，需要在两个组件之间同步SSL和TCP会话的状态.(仍互不信任)

③需对Android的默认SSL库做修改.

▲设计(分为两个部分：control plane和data plane)

control plane：控制数据流动的一组规则

data plane：根据规则数据流动的机制

→SplitPass control plane的策略存储在三个表格中：

LPT(local password table)和RRT(redirect rule table)存储在移动设备上

CPT(cloud password table)存储在辅助云上.

1>Data Plane：Cooperative Login.

▲移动设备需识别拼分割口令，将包含placeholder的网络包重定向到辅助云。后者首先检查一下目标地址，使用真实的"半口令组装成包，并发送给服务器

①SplitPass通过直接发送带placeholder的包完成TCP状态同步，使用packet filter将包重定向到辅助云，另外，SSL layer也发送必要的元数据到辅助云，包括server的IP地址，口令ID和SSL内部信息(如会话密钥和加密方法)

②辅助云收到重定向的包和元数据，包括口令ID用于找到另一部分口令，并使元数据中的敏感加密，将目的地址修改为服务器并发送包.(源地址仍是移动设备，因此对于server是无变化的)

③使每个SSL记录有个显式的IV，因此不需要同步.

2>Control Plane：Initialization.

辅助云使用4个API用于对口令配对和操作，在移动设备上使用app做配置并存储在table中

①4个API包括：CREATE_TABLE(), LIST_TABLE(), NEW_PASS(), DELETE_PASS().

②可以利用TrustZone保证用户和辅助云通信的隐私和完整性.

3>Whitelist-Based Server Authentication：为每一个的half-password设置IP白名单.

▲安全性分析

1>登录移动设备完全被腐化：A可获取设备中的口令份额和证书，并尝试从设备对辅助云攻击

①A欺骗辅助云发送口令份额到攻击者选控制的server，由于白名单机制，无法成功

②恶意修改操作表进行DoS攻击，本文使用备份到云的方式防止

2>云端完全被腐化：A可获取云端的口令份额和会话key.

但云端只接受移动设备的命令，不会直接发送命令，故无法用于攻击移动设备