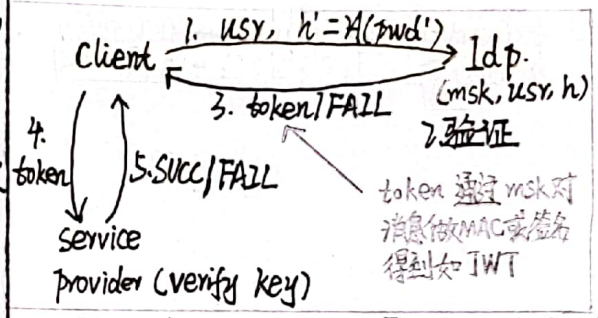
CCS'18- Agrawal- PASTA: PASsword-based Threshold Authentication

概述：(S)针对SSO应用场景中单个Idp面临单点故障的问题导致msk及用户口令的存储值泄漏的场景(I)本文提出了Password-based Threshold Authentication解决单个server breach的问题，(A)①作者基于game对多个client共用1个Idp的情景下的PbTA进行了形式化定义，并且使用threshold oblivious pseudo random function(TOPRF)和threshold token generation(TTG)作为构建块提出通用框架PASTA，构建2-round的协议，server之间无需交互，且token是否匹配在客户端验证。②作者实现了4种TTG方案(block-cipher based和DDH based threshold MAC. threshold RSA-based signature和threshold pairing based signature并分析了性能和安全性。(R)①作者证明了方案的安全性，开销是合理的，且证明了公钥操作不可避免.

优点：① 以game的形式定义不同方案的安全属性，比较清晰.
② 以"若可以以黑盒的方式利用单向函数构造PbTA，则可以只使用黑盒利用单向函数构建安全的密钥协商协议"的方式证明公钥操作不可避免.
③ 首次以4种方式实现5方案并进行了性能比较.

问题：① 从性能对比来看，客户端比服务器端的开销大一些，个人觉得在设计过程中，将token的合并与验证放在客户端
② 客户端需要自行生成PRF的key并注册到server中，需要客户端的改动
③ 本文假设口令均匀分布(虽然提到可扩展到其它情景).
④ 应讨论server被破坏后如何更新key

大多数software-based秘密管理系统使用token作为认证客户端的主要方法
JWT、SAML、OAuth、OIDC及kerberos等均使用类似原理，多用于enterprise环境中



Idp → single point of failure
故手可以：① 恢复msk伪造token的问资源
② 获取口令的存储值做离线字典攻击

↑研究背景：Idp server breach in password-based token generation scheme

▲Idea：构建Password-based Threshold Authentication (PbTA)，利用门限密码协议提供强有力的属性：unforgeability和password-safety

▲思路设计(要解决的关键问题)

a plain password-based token generation → 无法抵抗server breach

↓

①Threshold solution ⟹ 通过与TTG方案结合，msk由n个server分享，但S会存储口令的验证值(仅由口令计算得到)，若1个S被腐化，则会遭受离线字典攻击

↓

②更改存储在服务器中的信息 → 由TOPRF使存储的验证值同样由S决定，TOPRF中伪随机函数的k同样存储在n个server中，$h=F_k(pwd)$需t个S才可计算.

↓

③从four-round到two round → TOPRF需2-rounds发送和验证需2-rounds，如何解决？S直接生成token shares，使用h加密后发送给C解密得到token

↓

④破解客户端伪装攻击 → 而若h存储于server，则获取了h的故手可以伪装成C，此时将h分成n份存储在n个S中，6破坏1个server故手只能获取1个份额.

↓

⑤Multi-client 安全性 → 由于TOPRF的key对于client来说不变，故故手可以通过伪装成多个client得到所有PRF的值，因此让每个client自己生成TOPRF的key并在注册阶段在n个server中共享.

最终：U: user, password， C: PRF的key得token shares， S: msk和h的份额

▲PASTA通过组合TTG和TOPRF，以黑盒的方式构建PbTA方案，提供password-safety和unforgeability

1、安全假设：Gap Threshold One-More Diffie-Hellman (Gap-TOMDH) - Jarecki-ACNS'17

2、关键建技术组件及算法

1) TTG：将生成认证token的任务下发给n个server，提供强的unforgeability保证

4个相关多项式算法满足Consistency

若与好t-七个server交互，则不可能生成合法的token

① Setup(1^k, n, t) 生成sk及不同share，验证key vk和公共参数pp → (Π_{sk}, vk, pp)
② PartEval(sk_i, x) → y_i，通过sk_i和x计算token的share ③ Combine($\{i, y_i\}_{i\in S}$) =: tk/⊥ 将收到的token的share组合成token tk，失败返回⊥ ④ Verify(vk, x, tk) =: 1/0 通过vk验证tk是否有效

27 TOPRF - Jarecki ACNS'17 满足 unpredictability 和 obliviousness

(X, R)-TOP 包含4个PPT算法，满足 consistency

① Setup(1^k, n, t) → ({sk_i}, pp). 生成 sk 的 share 及公共参数 pp

② Encode(x, ρ) =: c. 使用随机值 ρ ER 编码 x∈X 得 c

③ Eval(sk_i, c) =: z_i 生成 TOPRF 的 share → 记为 TOP(sk, x)

④ Combine(x, {(i, z_i)}_{i∈S}, ρ) =: h/⊥ 使用 ρ 将 share 生成值 h 失败则生成 ⊥

PubCombine 由只能访问部分 Eval() 的实体检测给定的集合 S 是否生成对应的 PRF 值 (无法得全部 share)

例如 Z := {(i, z_i)}_{i∈S} 通过 consistency 产生，对于任意 Z* := {(i, z_i*)}_{i∈S}，若 PubCombine(Z) = PubCombine(Z*)，则有 Combine(x, Z, ρ) = Combine(x, Z*, ρ). 若前者不成立，则后者大概率不成立

TOPRF 的安全属性

① $Pr[Unpredictability_{TOP,Adv}(1^k, n, t) = 1] \le \frac{MAX_{|u|=t}(q_1, \cdots q_n)}{|X|} + negl(k)$ 难以被预测

由于敌手未知 P，因此敌手若不访问 Eval() 直接得到 TOPRF 输出的概率可忽略不记，但敌手可以对任意值求 Eval，并进行验证，若猜测 x 成功则增加成功的概率

② $Pr[obliviousness_{TOP,Adv}(1^k, n, t) = 1] \le \frac{MAX_{|u|=t}(q_1, \cdots q_n)+1}{|X|} + negl(k)$ 给定输出也难以猜测

与上述①不同，允许输出 TOPRF 的值，敌手可以在猜测失败再尝试1次.

PoTA, n个server及多个client，包含7个PPT算法，属于n阶段

1. global set-up phase n个shares

GlobalSetup(1^k, n, t, P) → ({sk_i}, vk, pp)，S_i收到(sk_i, pp)，初始化记录 REC_i := ∅

2. Registration Phase C不允许敌手交互，在安全信道进行.

① SignUp(C, pwd) → {(C, msg_i)}_{i∈[n]} 为每个server产出1个消息 msg_i

② Store(C, msg_i) =: rec_{i,C} 生成记录 rec_{i,C}，若不在C的记录则S_i存储(C, rec_{i,C})

3. Sign-on Phase (敌手可参与)

① Request(C, pwd, x, T) → (st, {(C, x, req_i)}_{i∈T}) 输出 secret state st 及请求消息 {req_i}_{i∈T}，将 (C, x, req_i) 发送给 S_i.

② Respond(sk_i, REC_i (record set), C, x, req_i) → res_i，输出响应信息 res_i

③ Finalize(st, {res_i}_{i∈T}) =: tk，输出 token.

4. Verification Phase

① Verify(vk, C, x, tk) → {0, 1}，输出是否为有效的 token.

▲安全属性(图7基于game的形式化描述清晰)

① Password Safety

$Pr[C^* \notin \gamma \wedge out = PwdList[C^*] \neq \perp] \le \frac{MAX_{|u|=t}(Q_{C^*,1}, \cdots, Q_{C^*,n})+1}{|P|} + negl(k)$

Adv可能猜测口令，生成请求，得响应得到 token，并验证 token 是否有效

② Unforgeability

若 $Q_{C^*}, x^* < \frac{1}{2}-|u|$，$Pr[Verify(vk, C^*, x^*, tk^*) = 0] \le negl(k)$

否则：$Pr[C^* \notin \gamma \wedge tk^* \notin TokList \wedge Verify(vk, C^*, x^*, tk^*) = 1] \le \frac{MAX_{|u|=t}(Q_{C^*,1}, \cdots, Q_{C^*,n})}{|P|} + negl(k)$

C未被corrupt且与足够多server交互，则敌手猜测正确口令不可以.

PASTA 的构造，包含 TTG, TOP, SKE(Encrypt, Decrypt)，H

TOP的输出用于加密 TTG 的 share，除此之外两者独立执行.

1. GlobalSetup(1^k, n, t, P) → ({sk_i}, vk, pp).

① TTG.Setup(1^k, n, t) → ({tsk_i}, tvk, tpp)

② sk_i := tsk_i, vk := tvk, pp := (k, n, t, P, tpp)

2. SignUp(C, pwd) → ((C, msg_1), ⋯, (C, msg_n)).

① TOP.Setup(1^k, n, t) → ({k_i}, opp).

② h := TOP(k, pwd), h_i = H(h||i), i∈[n]

③ 令 msg_i := (k_i, h_i), i∈[n]

注册

3. Store(sk_i, C, msg_i) =: rec_{i,C}.

① 解析 msg_i → (k_i, h_i) ② 令 rec_{i,C} := (k_i, h_i)

4. Request(C, pwd, x, T) → ({(C, x, req_i)}_{i∈T}, st)

① 若 |T| < t，输出⊥ ② ρ ←$ R, TOP.Encode(pwd, ρ) → c

③ 令 req_i := c, i∈[n], st := (C, pwd, ρ, T).

5. Respond(sk_i, REC_i, C, x, req_i) → res_i

① 若 rec_{i,C} ∉ REC_i，输出1，否则解析成 (k_i, h_i)

② TOP.Eval(k_i, req_i) → z_i,

③ TTG.PartEval(tsk_i, C||x) → y_i

④ 令 res_i := (z_i, SkE.Encrypt(h_i, y_i)).

6. Finalize(st, {res_i}_{i∈S}) → tk.

① 解析 res_i → (z_i, ctxt_i), st → (C, pwd, ρ, T)

② 若 S≠T，输出⊥

③ TOP.Combine(pwd, {(i, z_i)}_{i∈T}, ρ) → h.

④ 对于 i∈T, h_i := H(h||i), y_i := SkE.Decrypt(h_i, ctxt_i)

⑤ 令 tk → TTG.Combine({i, y_i}_{i∈T}).

登录

7. Verify(vk, C, x, tk) → {0, 1}.

① 输出 TTG.Verify(tvk, C||x, tk)