20/12/21

Mobile devices resilient to capture

(MacKenzie & Reiter 2003)

evitare il furto dei dati da PC/tel

Some statistics (@2010)

➔ **1 laptop stolen every 12 seconds**

➔ **Survey on 329 organizations:**

- ⇨ 86400 laptop lost; value: 2.1 B$
  - →Data value is greater than PC value!
- ⇨ 43% stolen at home

➔ **FBI data (2001-2005)**

- ⇨ 160 stolen laptop
- ⇨ 10+ with critical (classified) data
- ⇨ 51: unsure whether contained classified data

➔ **Famous R&B singer, Ryan Leslie**

- ⇨ Offered 1M$ for his stolen laptop in 2010
- ⇨ New original songs and videos worth A LOT for him
  - →(did not get it back)

Capture resilient device

➔ **A device that CANNOT be used by other than the rightful owner**

➔ **Assume "core" of the device is a secret key** tutto dipende da lei

- ⇨ E.g. to permit decryption, digital signatures, etc
- ⇨ E.g. SIM card

➔ **Possible security approaches**

- ⇨ Lock/Unlock key via passwd (e.g. cipher/decipher it)
  - →Weak: dictionary attack likely to succeed
- ⇨ Store secret in tamper-proof HW box (hardware costoso!)
  - →Must have it! Must be robust to side channel analysis
- ⇨ Dynamically download key from network repository
  - →Must trust network repository! uso provider! (usato oggi)

➔ **Other ideas?**

MacKenzie + Reiter, 2003

➔ **Assumption: device is connected when used**

➔ **Solution: involve a "capture-protection server" in the network**

- ⇨ Server confirms that device remains in owner's possession before permitting usage of key
- ⇨ SW only, no tamper-proof requirements

➔ **"capture-protection server" does NOT need to be trusted!**

➔ **Two approaches:**

- ⇨ Basic one, standard protocol
- ⇨ Extended one, uses (2,2) secret sharing
  - →Even if attacker cracks the device (and the user password), device key can be disabled

Scenario
(MacKenzie & Reiter)

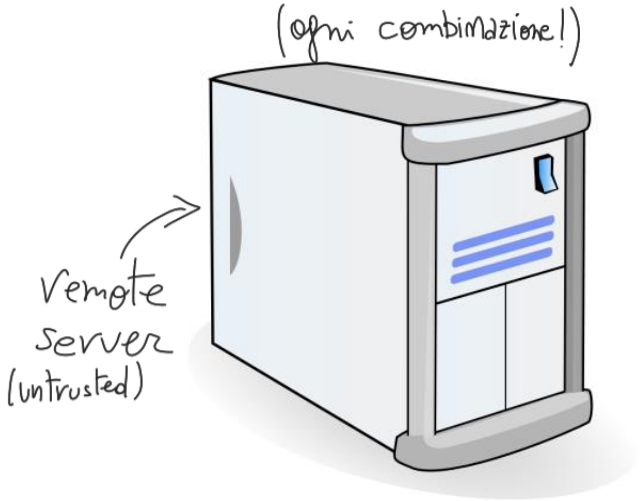
Password P

Many possible attack scenarios

- **password known**
- **device stolen**
- **server cracked (no trusted server!)**

Good: Solution resilient to any crack

Better: Solution resilient to more than 1 crack!



Secret Key	SKd
Public Key	PKd

Secret Key	SKs
Public Key	PKs

Basic solution

➔**Robust to following attacks:**

- ⇨Server cracked AND password known
  - →Attacker cannot sign/decrypt
- ⇨Device cracked/stolen
  - →Attacker can only perform dictionary attack ONLINE
- ⇨Device and Server cracked
  - →Attacker can only perform dictionary attack OFFLINE

➔**Basic solution NOT robust to:**

- ⇨Device AND password cracked
  - →Game over… Attacker can do all

Idea: use "tickets"

➔ **Assumption:**

⇨ device has access to network

➔ **Protection:**

⇨ Encrypt device key so that it can be decrypted only via cooperation with server

non mando chiave (server non fidato!) bensì tickets!

➔ **Idea:**

⇨ Send encrypted "ticket" to server

→Contains authentication material for user!!

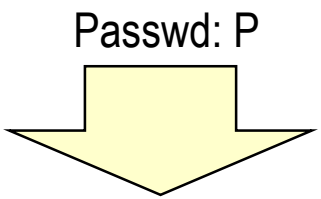
→No need to store on server

⇨ Use content of ticket to authenticate user

⇨ Use content of ticket to "partially" decrypt device key

→Final decryption at User – server never sees key!

Protocol: device initialization



Non da imparare, è più a scopo informativo, è complesso!
Capire sì, non memorizzare

Password (stored nel device) ① by User → true	P
	② PK_{server} (pubblica, eliminabile)
Gen random key	v (by PC, ④)
Gen random key	a
Hash password (computo)	b = H(P), deriva da P, è true!
Crypt Secret Key	c = $Sk_{device} \oplus PRF(v; P)$ (tolgo, encrypted, solo server può sapere!)
Create ticket	tkt = $E_{pk\text{-}server}[a,b,c]$ (encrypt)
Device public key	PK_{device} ③
Device Secret key	SK_{device} ③

Server Public Key

- PRF è keystream
- P by user, v by PC
- no secret sharing

Delete all red ⑤

se cifro qualcosa con input da user diversi ⇒ necessiterò di tali input dopo!

- a : random
- b : Hash(real PW)
- c : Crypt secret key

- tkt mantenuto localmente, No nel server, solo dopo il server può decriptarlo (SOLO LUI!)

SK_{device} da proteggere!!

Con i valori rimasti, senza b = H(P) NO dictionary attack!!

Protocol: key retrieval

(se sono veramente io e sono collegato ad internet, voglio accedere!)

Device

inserisco P

random key	v
random key	a (solo se rubano PC la hanno (se non sono io))
ticket	tkt = $E_{pk\text{-}server}[a,b,c]$ (solo server lo apre!)
Device public key	PK_{device}

se sono veramente io nel ticket ci sono loro, ma non posso ancora vederli!

b=H(P)

$c = Sk_{device} \oplus PRF(v; P)$

(non sa nulla, non ha dati sensibili è solo "assistente")

Server

User input password P

Compute hash $\beta=H(P)$ (localmente)

Gen random ρ (for encryption, non ci interessa)

send $mac_{aL}[tkt , E_{pk\text{-}server}[\beta, \rho]]$ (è uguale a quella nel ticket?)

estraggo

tkt	→ a,b,c
a (msg autentico?)	→ check MAC (auth device!) (↳ dice nulla sull'user)
decrypt	→ β, ρ
Check	→ b = β (auth user!)

$\rho \oplus c$ (≈ key transfer) return

protegge TRANSPORT

protegge STORAGE

Decrypt key

$(\rho \oplus c) \oplus \rho \oplus PRF[v,P] \rightarrow Sk_{device}$

da conoscere !!

Se rubano PC + pw RIP ✝

Attacks

➔ **Server cracked AND password known**

⇨ Key v in device, still secret

➔SK encrypted with PFR(v,passwd)

➔SK cannot be obtained

➔ **Device cracked/stolen**

⇨ Attacker must send valid passwd hash

⇨ can only perform dictionary attack ONLINE

la pw e' inviata online al server!

➔Easy to detect!

➔MAC verified (attacker knows a), but passwd fails many times

➔ **Device and Server cracked**

⇨ Passwd still missing

⇨ Dictionary attack OFFLINE against b=H(Passwd)

↑espongo lui

Can we do better?

➔**Limitation**

⇨device stolen and passwd known

➔Attacket can get SK and use it from now on

➔**Solution:**

⇨Must NOT reveal SK to the device itself!

⇨Easy (now ☺): use secret sharing!

➔**Following example:**

⇨RSA signature

⇨Analogous for encryption

Secret sharing (2,2) for RSA

$n = p \cdot q$

$d \leftarrow$ random, secret key

integer value, no problemi col modulo.

$d_1 \leftarrow$ random, share 1

$d_2 = d - d_1 \bmod \phi(n) \leftarrow$ share 2

NO lagrange

$$H(m)^{d_1} \cdot H(m)^{d_2} = H(m)^{d_1+d_2} = H(m)^{d_1+d-d_1} = H(m)^{d} \bmod n$$

Case (2,2): use trivial secret share

NO Shoup's problem to overcome in reconstruction (no Lagrange now!!)

Protocol2: device initialization

Password	P
Gen random key	v
Gen random key	a
Gen share 1	d_1=PRF(v; P)
Gen share 2	d_2=d-d_1 mod ϕ(n)
Gen disabling key	t
Gen disabling val	u=H(t)
Hash password	b = H(P)
~~Crypt Secret Key~~	~~c = Sk_{device} $\oplus$ PRF(v; P)~~
Create ticket	tkt = $E_{pk\text{-}server}$[a,b,u,d_2,N]
Device public RSA key (N,e)	
Device Secret RSA key d [Sk] [and ϕ(N)]	

PK_{server}

Server Public Key

core tecniche

NOTO nell'inizializzazione

① ②

device auth · PW auth · tecnica · 2° share · mod N

NOTO LUI, HO SK

- PRIMA: SK⊕PRF(V,P)
- ora creo share pseudo-RAND: devo avere V,P per computare, quando accedo COMPUTO il mio share, cioè ø info, al massimo sono uno dei due players!

Delete all red & orange

cose cambiate rispetto prima

per RSA signature, mando l'ALTRO share, serve anche il mio!

Protocol2: key retrieval

Device – while signing m

random key	v
random key	a
disabling key	t
ticket	tkt = $E_{pk\text{-}server}$[a,b,u,d_2,N]
Device public key	(e,N)

reference

b=H(P)

d_1 = PRF(v; P)

Se rubo device + pw
ho v, p → d_1 = PRF(v,p)
ma manca d_2 mai comunicato!!
Come capisce il server se device è rubato?
NON PUÒ. Uso key disabling

User input password P in device

Compute hash	β=H(P)
Gen random	ρ

computato dal device

msg da signare: mac_a[tkt , $E_{pk\text{-}server}$[H[m], β, ρ]]

H(pw)

send → (A transazione ho interazione col server! No solo per device activation!)

Server

tkt decrypt	→ a,b,d_2,u,N
a check	→ check MAC (auth device!)
decrypt	→ H[m], β, ρ
Check	→ b = β (auth user!)

signed with share

ρ ⊕ $H[m]^{d2}$

← **return**

Decrypt key

(ρ ⊕ $H[m]^{d2}$) ⊕ ρ

Complete signature

d_1 = PRF[v,P]

$H[m]^{d2}$ $H[m]^{d1}$ mod N

mai usata Sk. Prima sign con shares, poi ricostruisco signature (threshold enc)

Signature key d NEVER retrieved in clear!

If attacker gets passwd AND device, cannot get key d

Protocol2: key disabling

Comunico al server che mi hanno rubato il PC (uso Key t). $U = H(t)$, sta nel ticket da controllare, per bloccare signature

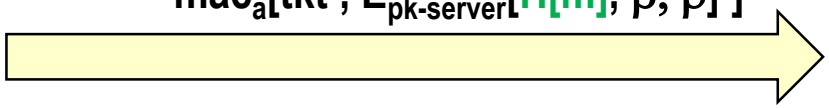
➔**Suffices to keep backup of**

⇨t

⇨tkt

➔**If device stolen, send them to server**

⇨Server keeps blacklist



Server

tkt → a,b,d_2,u,N

a → check MAC

(auth device!)

Search t for index tkt (e.g. hash table)

check u ≠ H(t) for corresponding t

(check if not revoked!)

Abbiamo visto sistema distribuito con diversi livelli di sicurezza!

- in RSA: threshold easy (tutto diretto / secret sharing)
- in ECDSA: harder!