Can ``Something You Know'' be Saved?

Baris Coskun and Cormac Herley
Polytechnic University, NY
Microsoft Research, Redmond

Introduction: Authentication

1. Something you know
   - Passwords, Passphrase
   - Challenge-Response
   - Graphical Passwords
   - Secret Questions

2. Something you have
   - RSA securID
   - smartcard

3. Something you are
   - Fingerprint, biometrics

Two factor just means two of the above:
e.g. Password + Smartcard

Challenge Response

- Problem with passwords is replay:
  - "Prove" identity by revealing secret (password)
  - Do this on untrusted PC, and keylogger knows it too!
- Can we reveal only part of secret?
  - E.g. suppose I memorize 256 bits
  - At login server challenges: SHA1(secret x salt) = ???
  - Now keylogger learns nothing
- Except I can't memorize 256 bits, or do SHA1
  - Within constraints of human memory (40-80 bits), and calculating power what can we do?

Attack Model

- Attacker observes everything on PC
  - keystrokes, mouse-moves, screenshots, traffic
- Attacker observes several login sessions
  - E.g. login many times from same PC

Why Bother with this?

- Aren't passwords going to be replaced by…..
  - Tokens, securID, 2 factor?
  - Some web 2.0 thing I read about?
- Maybe, but
  - Need ``Something You Know'' (at least as 2nd factor)
  - Instantaneous, free, ubiquitous
  - Only thing worse than 29 passwords is 29 smartcards!

Related Work

- Weinshall [2006]
  - Proposed Challenge Response scheme
- Golle and Wagner [2007]
  - Demonstrate brute force break.
- Lei et al [2007]
  - New scheme (see Appendix for break)
- Pattern:
  - Author 1: “here’s a clever scheme”
  - Author2: “here’s how to break it”
- Is there a systematic problem with Challenge Response?

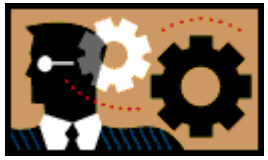
Challenge Response

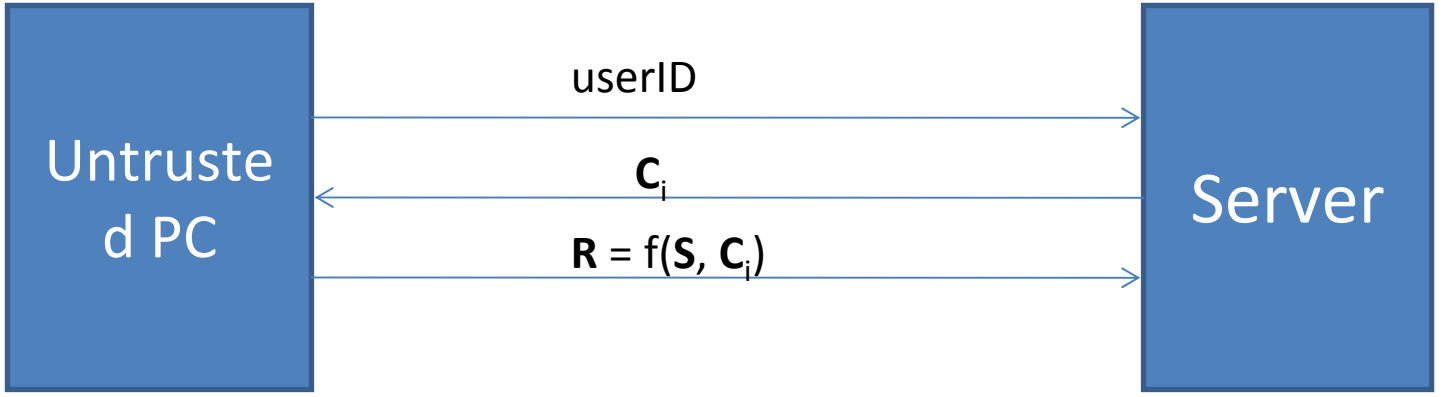
S	N-bit secret shared between user/server
$\mathbf{C}_i$	Challenge (random)
$\mathbf{R} = f(\mathbf{S}, \mathbf{C}_i)$	M-bit response (based on challenge and secret)
M	#bits of response > 20. Random guess has < 10^{-6} chance
N	#bits user must remember: TBD

Everything except **S** is public

User's task: remember N bits, perform calculation $\mathbf{R} = f(\mathbf{S}, \mathbf{C}_i)$, give M-bit response

Challenge Response





- User must calculate $\mathbf{R} = f(\mathbf{S}, \mathbf{C}_i)$ in his head
  - No use of untrusted PC

Example:

- Cryptographic Hash: f() = SHA1(), **S** > 256 bits
  - User returns **R** = SHA1(**S**, $\mathbf{C}_i$)
  - Problem: remember 256 bits, do SHA1 in head
- Challenge for random portions of secret
  - **S** = “Rex chewed Mary’s new slippers”
  - **C** = Deliver chars in posns 7, 9, 13, 17
  - **R** = “eeas”
  - Problem: attacker gets whole secret after few logins

A Single Login

- Response is M-bits (or M/k k-bit symbols)
  - $\mathbf{R} = f(\mathbf{S},\mathbf{C}_i) = \mathbf{R}_0\mathbf{R}_1\mathbf{R}_2 \ldots\ldots\mathbf{R}_{M/k-1}$
- How many bits of **S** involved in calculating $\mathbf{R}_i$?
- Suppose all N bits of **S** used for each bit of $\mathbf{R}_i$
  - Requires at least M(N-1) binary decisions
  - E.g. 20 (80-1) = 1580 decisions
  - User performs 2 decisions/second → 13.3 minutes!
- So only U << N bits involved in each symbol $\mathbf{R}_i$

Model

- W logins => MW-bit stream
  - $\Gamma = \underbrace{\mathbf{R}_0\mathbf{R}_1\mathbf{R}_2 \ldots\ldots\mathbf{R}_{M/k-1}}_{1^{st}\text{ M-bit login}}\ \underbrace{\mathbf{R}_{M/k}\mathbf{R}_{M/k+1} \ldots\ldots\mathbf{R}_{W-1}}_{W-1\text{ logins}}$

- Attacker can try many offline attempts
  - For each secret **S'** calculate
  - $\Gamma' = \mathbf{R'}_0\mathbf{R'}_1\mathbf{R'}_2 \ldots\ldots\mathbf{R'}_{M/k-1}\ \mathbf{R'}_{M/k}\mathbf{R'}_{M/k+1} \ldots\ldots\mathbf{R'}_{W-1}$
- If $\Gamma = \Gamma'$ attacker is done.

How Many bits of Secret involved in each output symbol

- Two secrets **S** and **S'** differ in e posns
- What about their responses?
  - $\mathbf{R}_0\mathbf{R}_1\mathbf{R}_2 \ldots\ldots\mathbf{R}_{M/k-1}$
  - $\mathbf{R'}_0\mathbf{R'}_1\mathbf{R'}_2 \ldots\ldots\mathbf{R'}_{M/k-1}$
- Only U<<N bits of **S** involved in each $\mathbf{R}_i$
- When e <<N high probability that none of the e bits where **S** and **S'** differ among U involved

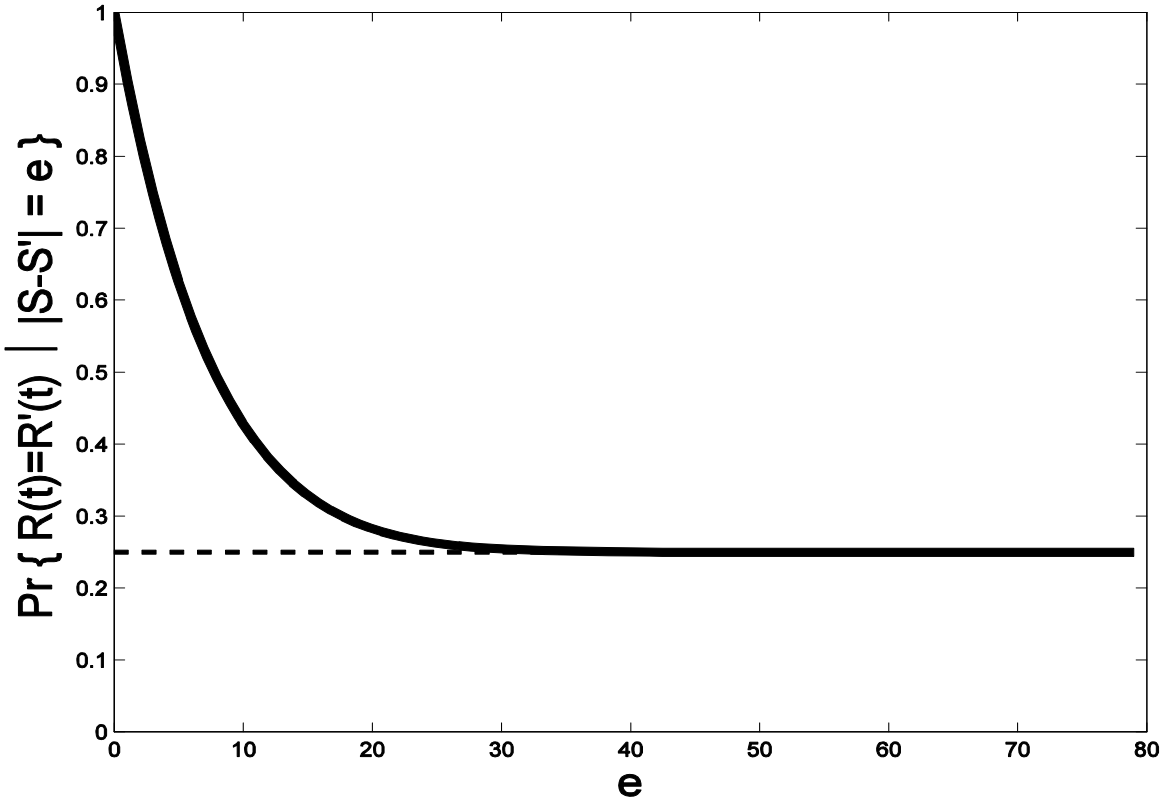
$$\mathbf{R}_i = \mathbf{R'}_l$$

A Generic Brute-Force Attack

1. When **S** and **S**’ are close **Γ** and **Γ**’ are close
2. It’s easy to find an **S**’ that’s close to **S**
3. Once close it’s easier to get closer

Secrets close => Responses close

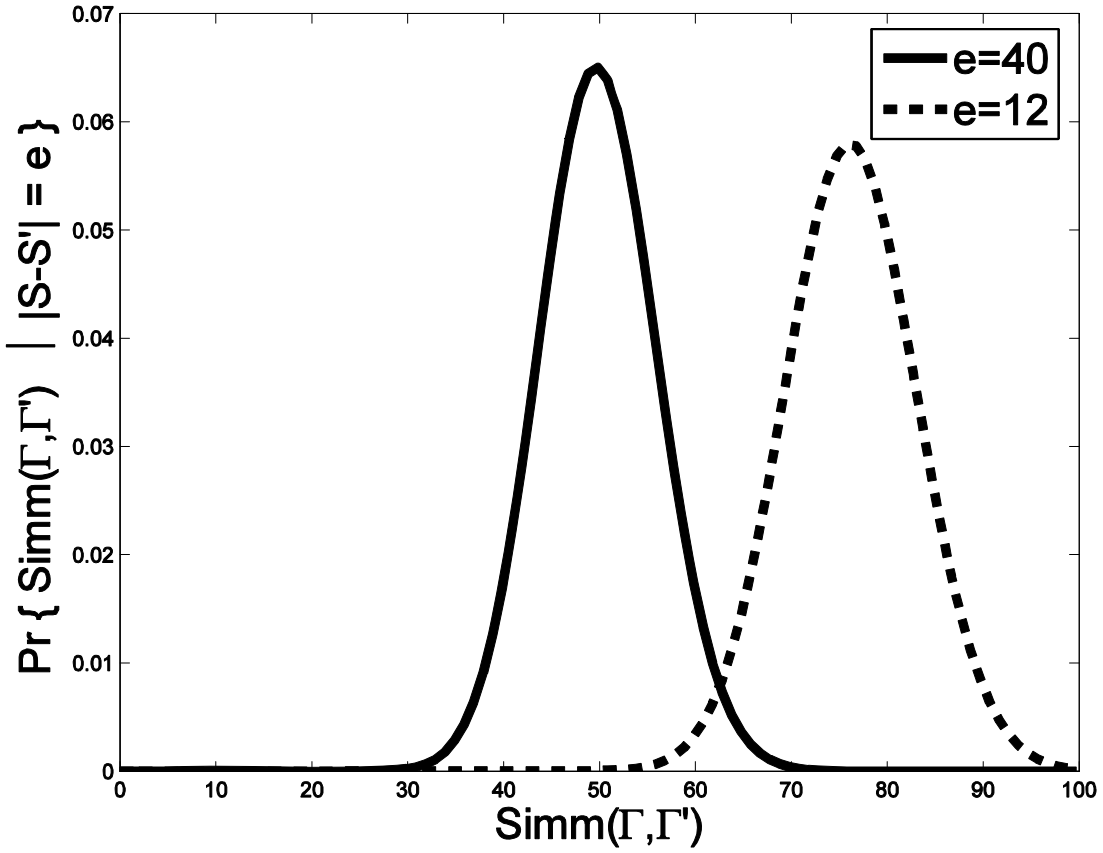
Prob{$\mathbf{R}_i$=$\mathbf{R'}_i$ given |**S**-**S'**|=e, N=80, U=10}



- |**S**-**S'**|large
  - Pr ~ 0.25
- |**S**-**S'**|small
  - Pr{} → 1

Secrets close => Responses close

Simm(Γ , Γ ') same given
|**S**-**S**'|=40 and 12; N=80, U=10, W=20



- Can distinguish
  - |**S**-**S**'| large
  - |**S**-**S**'|small
- **Responses coincide more**

Easy to get a secret that's close

- Enumerate **Γ'** for large number of secrets **S'**
- Retain those for which
  - Simm(**Γ**, **Γ'**) is large
- With high probability have at least one **S'**
  - |**S** – **S'**| is small

Once close, easy to get closer

- Suppose we're close:
  - $|\mathbf{S}\text{-}\mathbf{S}'|$=e and e is small
- Flip one bit of **S**':
  - Either distance e-1 or e+1
  - Distance e-1 produce responses more like $\mathbf{\Gamma}$ than distance e+1 neighbors
  - Repeat and iterate to **S**

The Generic Attack

- Choose enough secrets **S**’ to ensure that several are close to **S**
- Retain those where
  - Simm(**Γ,Γ’**) is large
- On all remaining secrets **S**’
  - Iterate to get closer.

What's needed to resist Brute-force?

- Time to Brute-force secret

#Logins	N=50	N=60	N=70	N=80
10	9.9	24	16	58
15	10.5	15.9	23	32
20	12.2	20.5	30.2	42
25	17.5	27.8	41.4	57

- Recall: user must do > 20(N-1) decisions

Conclusion

- If (Secrets close => Response close)
  - then GameOver
- When U << N scheme easily brute-forced
- If we cannot restrict #logins observed
  - Very hard to find anything between
    - Passwords
    - Tokens, securID, OTP's, 2 factor
- Questions?