Sistemi operativi e reti[1]

Mattia Monga

Dip. di Informatica
Università degli Studi di Milano, Italia
mattia.monga@unimi.it

a.a. 2014/15

Lezione I: Sicurezza informatica

Paradigmi di interazione in rete

Client-Server

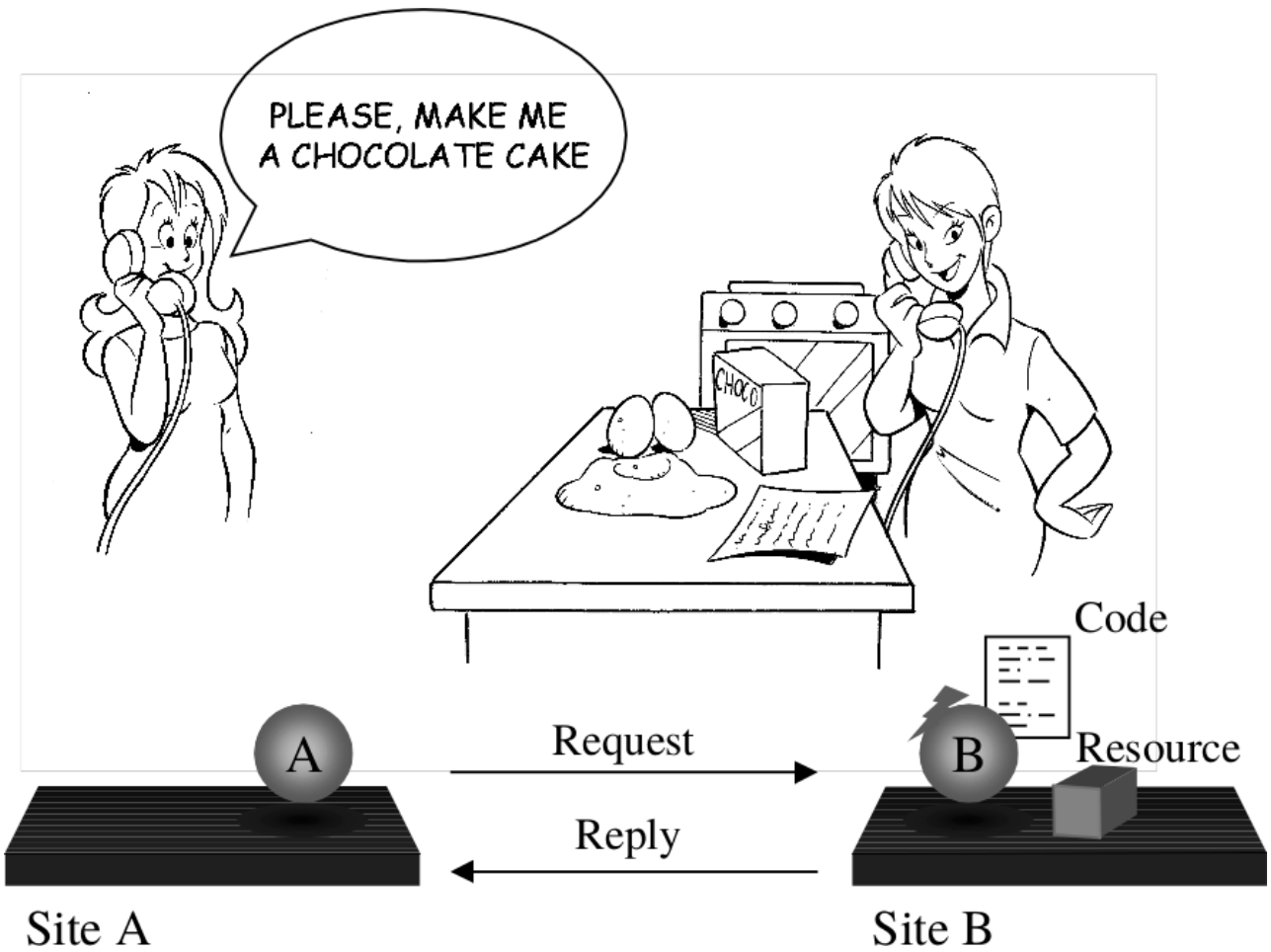
Remote Evaluation

Code on demand

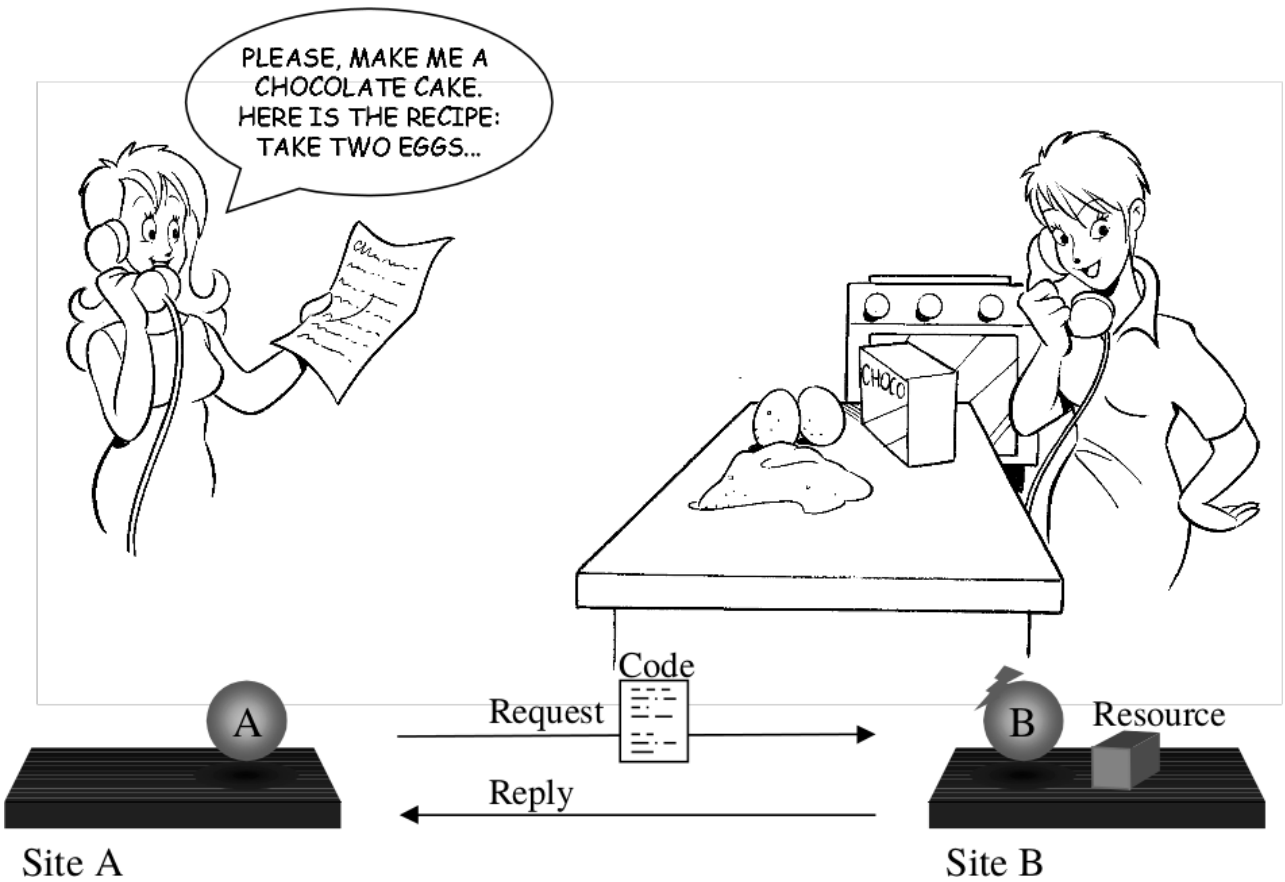
Mobile agents

Per fare un *torta* (output) servono: una *ricetta* (il programma), gli *ingredienti* (input) e un *forno* (mezzo di calcolo).
Figure da: G. Vigna, "Mobile Code Technologies, Paradigms, and Applications".

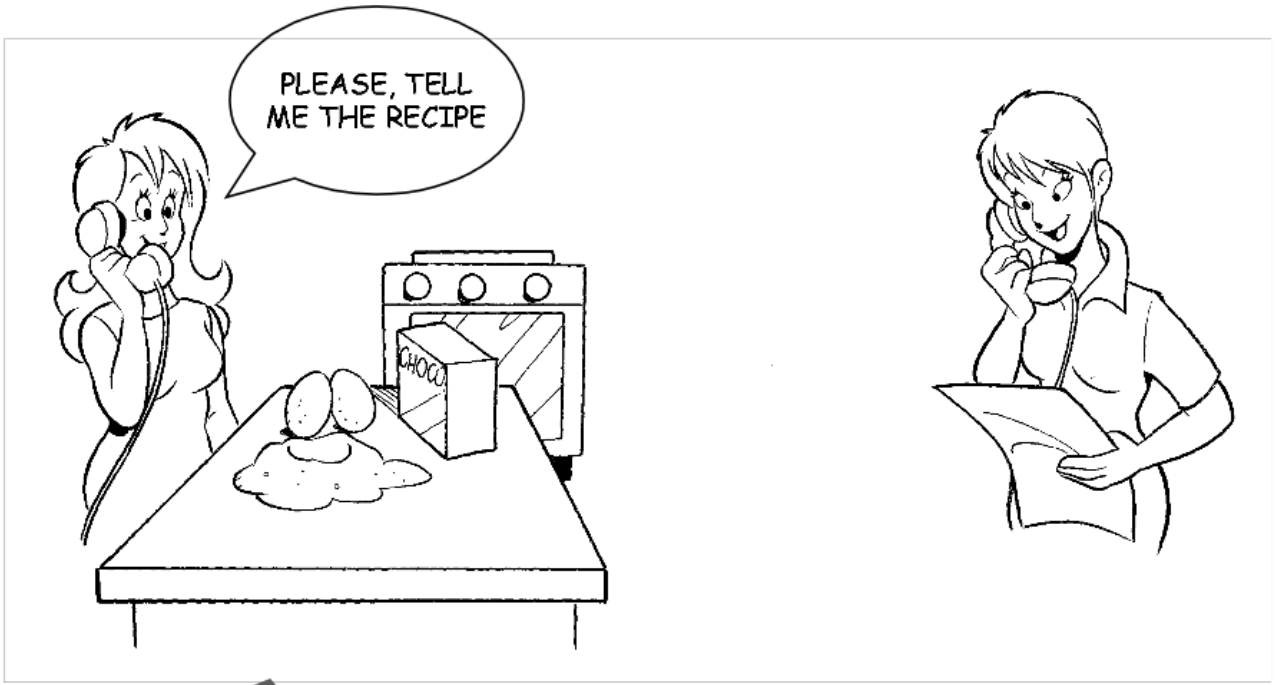
Client-Server



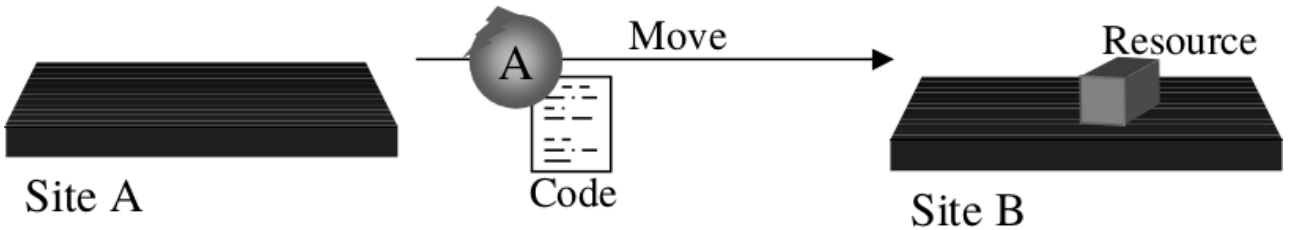
Remote evaluation



Code on Demand



Mobile Agents



Ma in rete è tutto virtuale. . .

Thomas Montgomery, 48 anni, di Buffalo, si è creato un'identità da diciottenne in una chat per rimorchiare. È riuscito a fare amicizia con un'altra diciottenne, ma poco dopo è stato scavalcato nel cuore della ragazza da Brian Barrett, 22 anni. A quel punto Montgomery, che ha due figlie ed è diacono della sua chiesa, ha scoperto l'indirizzo di Barrett e lo ha ucciso. Al processo si è scoperto che la ragazza, Mary Sheiler, aveva in realtà 50 anni e usava per il suo profilo online una foto della figlia. [Internazionale n. 728]

Cartoon by Peter Steiner. The New Yorker, July 5, 1993 issue (Vol.69, no. 20)

Le interazioni necessitano di fiducia

Come costruire la fiducia fra le parti?

Autenticare	identificare e verificare il possesso di un titolo d'autorizzazione
Autorizzare	permettere o vietare l'accesso a una risorsa
Proteggere	la conoscenza di informazioni anche in transito
Garantire	la tutela di diritti/servizi/beni acquistati

Cos'è la sicurezza?

Si parla di sicurezza quando c'è *qualcosa* (asset) da proteggere.

safety gli utenti di un sistema sono protetti rispetto a danni causati dal suo funzionamento

security le risorse (dati, funzionalità, business) di un sistema sono protette rispetto ad usi impropri

Security: Politiche di sicurezza

Le politiche di sicurezza (*policy*) stabiliscono quali siano gli usi appropriati e quelli impropri. In ambito informatico generalmente regolano

confidenzialità dati e servizi sono accessibili solo da chi è autorizzato

integrità dati e servizi possono essere alterati solo in secondo la politica

disponibilità dati e servizi rimangono disponibili secondo la politica

Costi e rischi

Si tratta di proprietà *non-funzionali* e impossibili da raggiungere in assoluto: occorre trovare un equilibrio fra:

costo delle misure di sicurezza

valore degli *asset*

rischio di attacchi

Cifratura

Per proteggere un'informazione (confidenzialità) si possono usare tecniche di *cifratura*

$$\texttt{testo in chiaro} \overset{\text{cifratura}}{\rightsquigarrow} \texttt{jAOEAwMCA6SVSID}$$

$$\texttt{jAOEAwMCA6SVSID} \overset{\text{decifratura}}{\rightsquigarrow} \texttt{testo in chiaro}$$

La cifratura e decifratura tipicamente sono algoritmi fissi (e pubblici), che agiscono in maniera diversa in base ad una o piú chiavi. Il principio di Kerckhoffs-Shannon: *il nemico prima o poi conoscerà il tuo sistema*; cambiare la chiave è piú facile che cambiare sistema!

La cifratura perfetta

La cifratura perfetta è stata inventata nel 1917 da J. Mauborgne e G. Vernam: One-time pad

- Si produce uno stream di bit completamente random (tirando una moneta?) della stessa lunghezza del testo da cifrare
- Si fa lo XOR bit a bit con il testo in chiaro

$0 \oplus 0 = 0$
$0 \oplus 1 = 1$
$1 \oplus 0 = 1$
$1 \oplus 1 = 0$
$011 \oplus 110 = 101$ $101 \oplus 110 = 011$
$(c \oplus x) \oplus x = c$

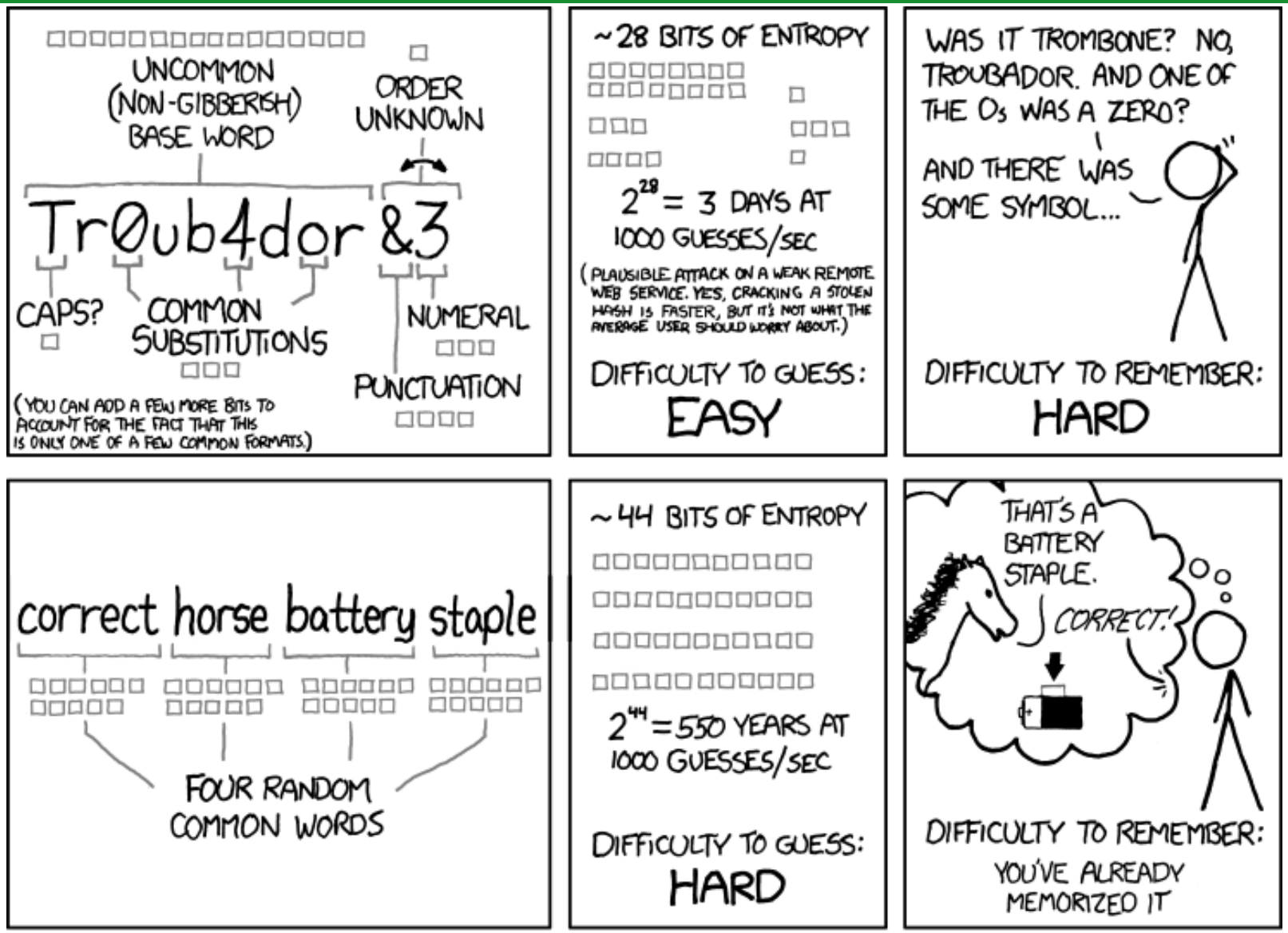
Problema: la chiave deve essere veramente imprevedibile usata una sola volta e **lunga quanto il messaggio**!

Password

La cifratura può essere usata per controllare l'accesso a una risorsa: una parola d'ordine (*chiave*) è un segreto condiviso (fra autorizzatore e autorizzando) che permette di controllare che un agente sia effettivamente autorizzato ad agire.
Il punto cruciale è che deve essere un segreto difficile da indovinare: solo chi è autorizzato è capace di fornire le credenziali corrette.

Password guessing

- Una password può essere scelta in maniera prevedibile (anziché del tutto casuale) nell'insieme possibile.
- *Online guessing*: l'attaccante prova tutte le password possibili (brute force); si limitano i tentativi e/o si rallenta il feedback


UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor&3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)
~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS:
EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER:
HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS:
HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER:
YOU'VE ALREADY MEMORIZED IT
THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS

	45	3
	182	4
	1002	5
	3106	6
	5694	7
	10748	8
	15374	9
	19126	10
	20532	11
	15996	12
	11225	13
Da `/usr/share/dict/italian`	6931	14
	3535	15
	2020	16
	733	17
	339	18
	160	19
	72	20
	40	21
	10	22
	2	23
	5	24
	1	25

$$26^8 = 2,088 \cdot 10^{11}$$

$$4000^4 = 2,560 \cdot 10^{14}$$

Capacità di “forza bruta”

Pentium 100: 10000 pwd/s
PC odierno: 100M pwd/s
Sforzi distribuiti: 80G pwd/s

Funzioni hash

Sono funzioni $h = H(x)$:

- non invertibili (per ogni h ci sono infiniti x)
- non esiste un metodo efficiente per trovare $\hat{x}$ per cui $H(\hat{x}) = \hat{h}$ per un $\hat{h}$ assegnato

Es: SHA256

Offline guessing

Offline guessing: l'attaccante accede all'elenco dei segreti (generalmente crittati con hash) e prova elenchi di parole (dictionary attack); si salano gli hash per rendere impraticabile la realizzazione di *rainbow table*.

Utente	salt	stored password
Alice	42	$\mathrm{hash}(42 \vert \mathrm{password}_{\mathrm{Alice}})$

- Possibilità di intercettazione
- Utilizzo in occasioni differenti
- distribuzione iniziale delle credenziali (si fanno scadere al primo accesso)

Digest

Le funzioni hash possono essere usate per controllare l'integrità di un'informazione

Lo schema di Lamport

Le funzioni di hash possono essere utilizzate per costruire schemi di "One Time Password" che non necessitano la necessità di sincronizzazione temporale (infatti è possibile usare un orologio e un segreto condiviso per generare password da usare una volta sola: ma tenere sincronizzati due orologi è un problema piuttosto difficile!).

Lo schema di Lamport (1981)

1. Alice e Bob concordano un segreto W
2. Bob conserva $H(\dots H(H(W))\dots) = H^n(W)$ e n
3. Autenticazione
   1. Alice comunica la propria *username*
   2. Bob risponde con il numero n
   3. Alice comunica $x = H^{n-1}(W)$
   4. Bob verifica che $H(x) = H^n(W)$ e decrementa n

Lo schema funziona n volte, poi bisogna cambiare W.

Canali sicuri

Il problema principale delle tecniche viste finora è la necessità di *condividere preventivamente un segreto*: serve un "canale" già "sicuro"...
I protocolli per lo scambio di chiavi e la crittografia asimmetrica nascono per risolvere/alleviare questo problema.
Radiodramma: "Biciclette e lucchetti"