Računarska bezbednost

Provere identiteta i lozinke

Petar Veličković

Matematička gimnazija, Nedelja informatike

1. april 2015.

Računarska bezbednost

- **Računarska bezbednost:** *disciplina koja se suprotstavlja neželjenim namerama i ponašanjima koja uključuju informacionu i komunikacionu tehnologiju.*

- **Security** i safety – u našem jeziku uglavnom sinonimi, ali prvi termin označava odbranu od **namernog** malicioznog ponašanja ($\sim$ inteligentan protivnik).

Provera identiteta

- Jedan od najosnovnijih mehanizama (računarske) bezbednosti: *dozvoliti neku akciju samo osobi koja je ovlašćena za tu akciju.*

- Zahteva jasno definisan protokol za **proveru identiteta** – glavna tema ovog predavanja (uz nekoliko skretanja).

Identifikacija ljudi

Ljudi se mogu identifikovati na više načina:

- **Nečim što jesu**

  Biometrijska identifikacija: otisak prstiju, dužica oka...

- **Nečim što rade**

  Rukopis, glas...

- **Nečim što imaju**

  Ključ, smart kartica, mobilni telefon...

- **Nečim što znaju**

  Lozinke, PINovi, tajna pitanja...

Osnovne kriptografske funkcije

- Nećemo se osvrtati na kriptografske protokole koji ovo čine mogućim; pretpostavka je da imamo pristup sledećim funkcijama:
  - Enkripcijska funkcija, $\text{Enc}_K(X)$; funkcija koja šifrira podatak X koristeći ključ K, tako da je **jako teško** doći do originalnog podatka bez znanja ključa.
  - Potpisna funkcija, $\text{Mac}_K(X)$; funkcija koja potvrdjuje integritet nekog podatka, tako da je **jako teško** generisati novi par $(Y,\ \text{Mac}_K(Y))$ bez znanja ključa.
  - Heš funkcija, $H(X)$; funkcija koja mapira podatke iz jedne forme u drugu, tako da je **jako teško** pronaći dva podatka x i x' tako da važi $x \neq x' \wedge H(x) = H(x')$.
- Jako teško – složenost $\approx 2^{80}$ operacija.

Kombinovanje kriptografskih funkcija

- Enkripcijska funkcija i potpisna funkcija se mogu kombinovati; ovim dobijamo i tajnost i integritet.
- Korisnik ima na raspolaganju funkcije Enc_{K_1} i Mac_{K_2}, i želi da napravi protokol takav da:
  - Može jednostavno poslati podatak X osobi koja zna ključeve K_1 i K_2;
  - Osoba koja ne zna ključ K_1 ne može saznati X ukoliko presretne poslati podatak;
  - Osoba koja ne zna ključ K_2 ne može generisati nove (lažno) šifrirane podatke.
- Jedan mogući način kombinovanja: korisnik šalje par $(\text{Enc}_{K_1}(X),\ \text{Mac}_{K_2}(\text{Enc}_{K_1}(X))$
  Označićemo ovaj par kao $\{X\}_K$ (gde je $K = (K_1,\ K_2)$)

Implementacija

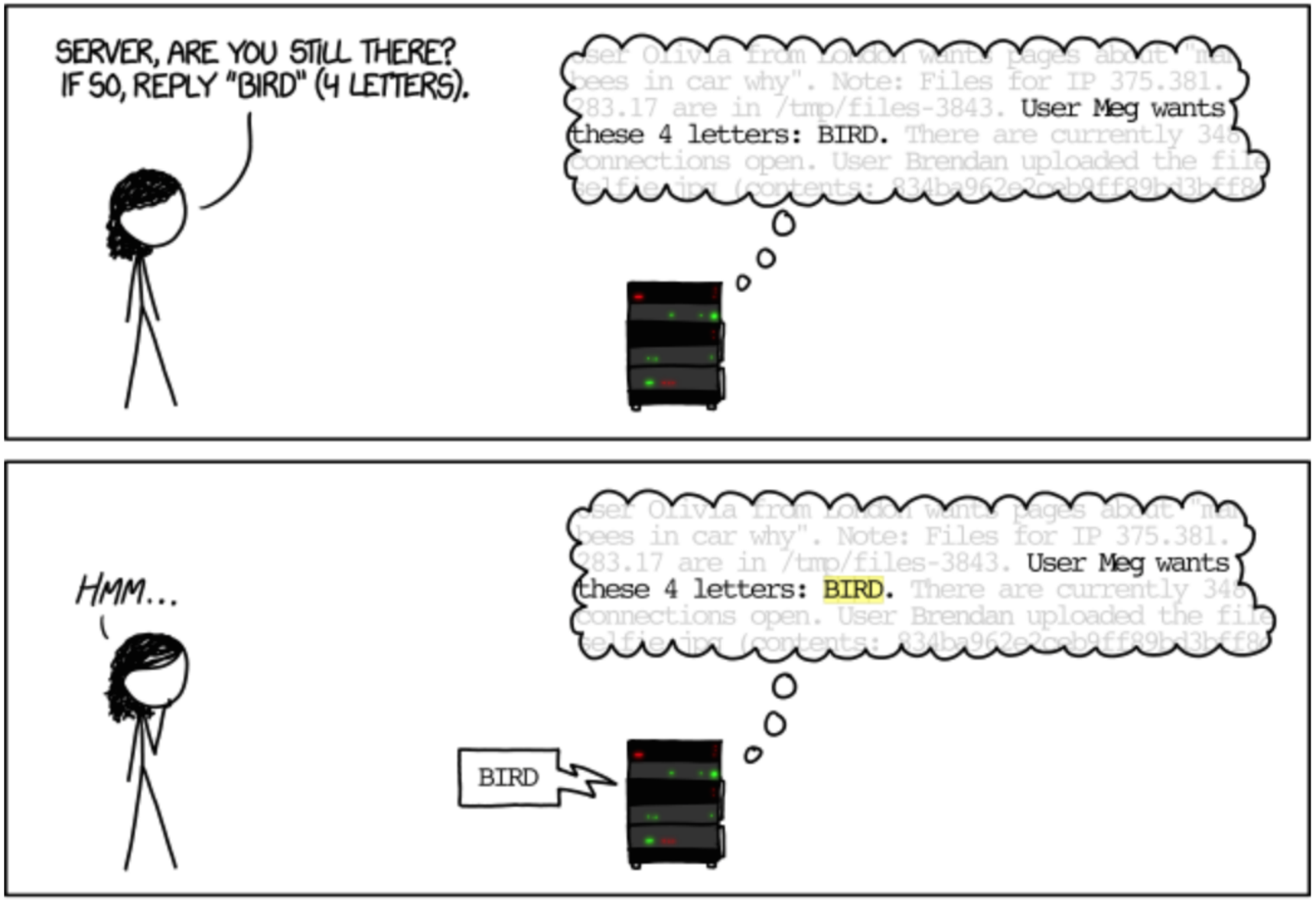
- Važno je zapamtiti: *svi, pa i najbolji kriptografski protokoli vrlo lako pokleknu ukoliko se loše implementiraju!*

- Skorašnji primer: *Heartbleed* — bag u OpenSSL implementaciji TLS protokola (kojim se šifriraju informacije na Internetu), otkriven u aprilu 2014; pogođeno oko 500.000 servera!

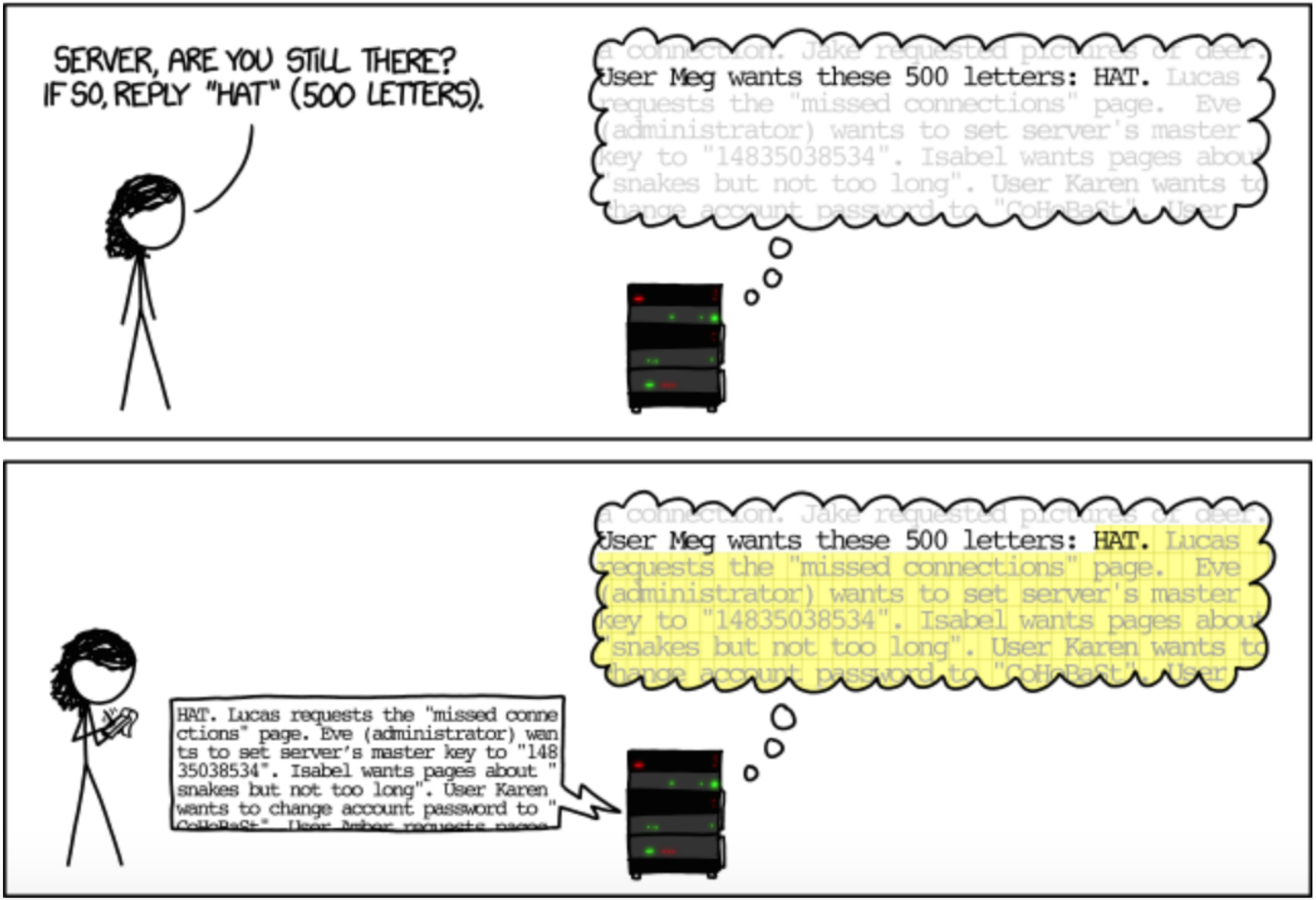
Heartbleed

- SSL/TLS protokol koristi 'heartbeat' poruke, da bi periodično proveravao da li je veza i dalje aktivna.
- Heartbeat zahtev se sastoji od nekog stringa S, i njegove dužine, L. Server koji primi heartbeat zahtev je u obavezi da vrati taj isti string pošiljaocu.
- OpenSSL nije proveravao da li je dužina ovog stringa zaista jednaka dužini koja je data u zahtevu, i samo je alocirao niz dužine L, koji bi popunio stringom S i vratio nazad.
- Ukoliko je $L > |S|$, ovo bi korisniku omogućilo direktan i neopažen pristup sadržaju radne memorije servera, u kojoj se mogu nalaziti tajni ključevi, lozinke...

Heartbleed Explanation (http://xkcd.com/1354/)



Heartbleed Explanation (http://xkcd.com/1354/)



Identifikacija pomoću deljenih tajni

- Glavni problem koji ćemo posmatrati:
  Korisnik (A) i server (S) zajedno dele neku tajnu (K_{as}); npr. lozinku tog korisnika.

- A želi da se identifikuje kod S, da bi mogao npr. da pristupi svom profilu na nekom sajtu.

- Notacija koju ćemo koristiti pri opisivanju protokola:

$A \to S : D$	(A je poslao serveru S podatak D)
$\mathcal{R}$	(nasumičan broj)
$X \| \| Y$	(spajanje podataka X i Y u jedan)
$\text{Mac}_K(X)$	(podatak X potpisan koristeći ključ K)
$\{X\}_K$	(podatak X šifriran i potpisan koristeći ključ K)

Jednostavni protokoli za identifikaciju

Lozinka:

$$A \rightarrow S : \quad K_{as}$$

Problemi:

- Lozinku je moguće presretnuti.
- S ne može potvrditi identitet A.
- Mnogo suptilniji problemi... (videćemo kasnije)

Jednostavni protokoli za identifikaciju

Jednosmerni izazov:

$$\begin{aligned} S \to A: &\quad \mathcal{R} \\ A \to S: &\quad \text{Mac}_{K_{as}}(\mathcal{R}) \end{aligned}$$

Dvosmerni izazov:

$$\begin{aligned} S \to A: &\quad \mathcal{R}_s \\ A \to S: &\quad \{\mathcal{R}_s || \mathcal{R}_a\}_{K_{as}} \\ S \to A: &\quad \mathcal{R}_a \end{aligned}$$

Jednostavni protokoli za identifikaciju

Jednokratna lozinka:

$$\begin{aligned} A \to S: &\quad C||\text{Mac}_{K_{as}}(C) \\ A: &\quad C := C + 1 \end{aligned}$$

- Server pamti poslednju lozinku C_{last} koju je prihvatio, i ne prihvata $C' \leq C_{last}$.
- Protokol često korišćen u ključevima od kola.

Jednostavni protokoli za identifikaciju

Master-ključ sistem:

- Server poseduje glavni ključ K, na osnovu kog generiše ključ K_i za svakog korisnika A_i.

$$\begin{aligned} A_i \to S: &\quad i \\ S \to A_i: &\quad \mathcal{R} \\ A_i \to S: &\quad \text{Mac}_{K_i}(\mathcal{R}) \end{aligned}$$

- Protokol često korišćen u smart karticama.

Napad na protokol

Podsetnik – **Dvosmerni izazov:**

$$\begin{aligned} S \to A: &\quad \mathcal{R}_s \\ A \to S: &\quad \{\mathcal{R}_s || \mathcal{R}_a\}_{K_{as}} \\ S \to A: &\quad \mathcal{R}_a \end{aligned}$$

Napadač E se može uspešno identifikovati kao A, tako što će paralelno zadati svoj izazov serveru (označen crvenom bojom):

$$S \to E: \quad \mathcal{R}_s$$

$$\begin{aligned} E \to S: &\quad \mathcal{R}_s \\ S \to E: &\quad \{\mathcal{R}_s || \mathcal{R}'_s\}_{K_{as}} \end{aligned}$$

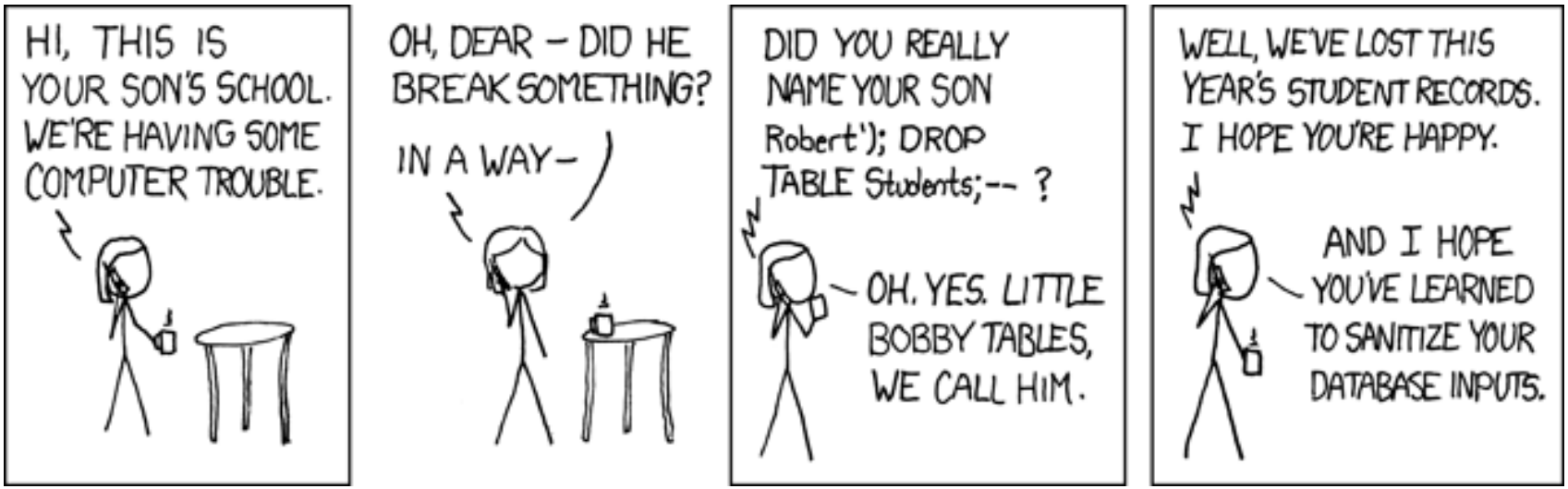
$$\begin{aligned} E \to S: &\quad \{\mathcal{R}_s || \mathcal{R}'_s\}_{K_{as}} \\ S \to E: &\quad \mathcal{R}'_s \end{aligned}$$

- Moguća ispravka: $K_{as} \neq K_{sa}$.
- *Nikad ne koristiti isti ključ za različite svrhe!*

Implementacija lozinki na serveru

- U ostatku predavanja ćemo se baviti specifično lozinkama.
- Kako čuvati lozinke na serveru?
- Najjednostavnija implementacija čuva lozinku u bazi podataka bez dalje izmene.
  - Vrlo nebezbedno, ali i pored toga **preko 30% sajtova trenutno funkcioniše na ovaj način!**
  - Većina takvih sajtova ne štiti propisno ni svoje baze podataka — ranjivi su SQL injectionu!

Exploits of a Mom (`http://xkcd.com/327/`)



strcmp napad

- Čak i ukoliko smo zaštitili bazu od SQL injectiona, postoje drugi načini kojim se može napasti sistem koji čuva čiste lozinke u bazi podataka.
- Na primer – sistem koji proverava tačnost lozinke tako što poziva strcmp funkciju:

```
1 int strcmp(const char *s1, const char *s2)
2 {
3     while (*s1 == *s2++)
4         if (*s1++ == 0)
5             return (0);
6     return (*(const unsigned char *)s1 - *(const unsigned
      char *)(s2 - 1));
7 }
```

`strcmp` napad

- Pokušati sva moguća slova kao prva slova u lozinci, i izmeriti vremena potrebna serveru da nam odgovori da je svaka lozinka pogrešna.
- Pretpostavimo da je provera lozinke sa prvim slovom `'s'` trajala $\approx 2\mu s$ duže od svih ostalih.
- Algoritam je napravio jednu iteraciju više $\implies$ `'s'` je prvo slovo lozinke!
- Slično nastaviti za ostala slova.

Implementacija lozinki na serveru – *hash*

- **Potencijalno rešenje:** umesto čuvanja lozinke P u bazi podataka, čuvati $H(P)$ (gde je H bezbedna heš funkcija)

- **Problemi:**
  - Baza podataka i dalje ranjiva; napadači mogu koristiti velike tabele sa preračunatim vrednostima heš funkcije za najčešće korišćene lozinke (*rainbow tables*).
  - Loš odabir heš funkcije: veliki procenat sajtova koristi *MD5* (dokazano probijen u $< 1s$ na normalnom računaru) ili *SHA-1* (očekivano probijanje u narednim godinama).
  - Rešenje: koristiti jaču funkciju, npr. *SHA-2*.

Implementacija lozinki na serveru – *hash + salt*

- **Rešenje:** Za lozinku P, čuvati $(S,\ H^n(S||P))$ u bazi podataka.
- S je nasumična vrednost (*"salt"*) koja se dodaje kao prefiks lozinci, nakon čega se heš funkcija primenjuje nekoliko puta.
- Ukoliko se koristi jaka heš funkcija, ovakva konstrukcija je (teoretski) bezbedna čak i ukoliko se baza podataka ukrade.
- Međutim, nevolje su tek počele...

Bezbednosna psihologija

- Primarni korisnici lozinki nisu programi, nego **ljudi**.
- Računarska bezbednost zahteva i obavezan element **psihologije i sociologije**.

Upotrebljivost bezbednosnih sistema

- Poznato je da ljudi uglavnom ne uspevaju da izađu na kraj sa bezbednosnim sistemima i protokolima.
  - Whitten, Tygar: *"Why Johnny Can't Encrypt"*, 1999.
  - Studija upotrebljivosti PGP protokola za šifriranje e-mailova.
  - Korisnici raznih zanimanja i nivoa obrazovanja, ali iskusni u primanju i slanju e-maila.
  - Dato im je 90 minuta da obave nekoliko svakodnevnih zadataka koristeći PGP radi zaštite sadržaja poruka.

- 90% korisnika nije uspelo! Štaviše, oko 40% se **potpuno kompromitovalo**, objavljujući svoje privatne ključeve zajedno sa ili umesto javnih.

Upotrebljivost lozinki

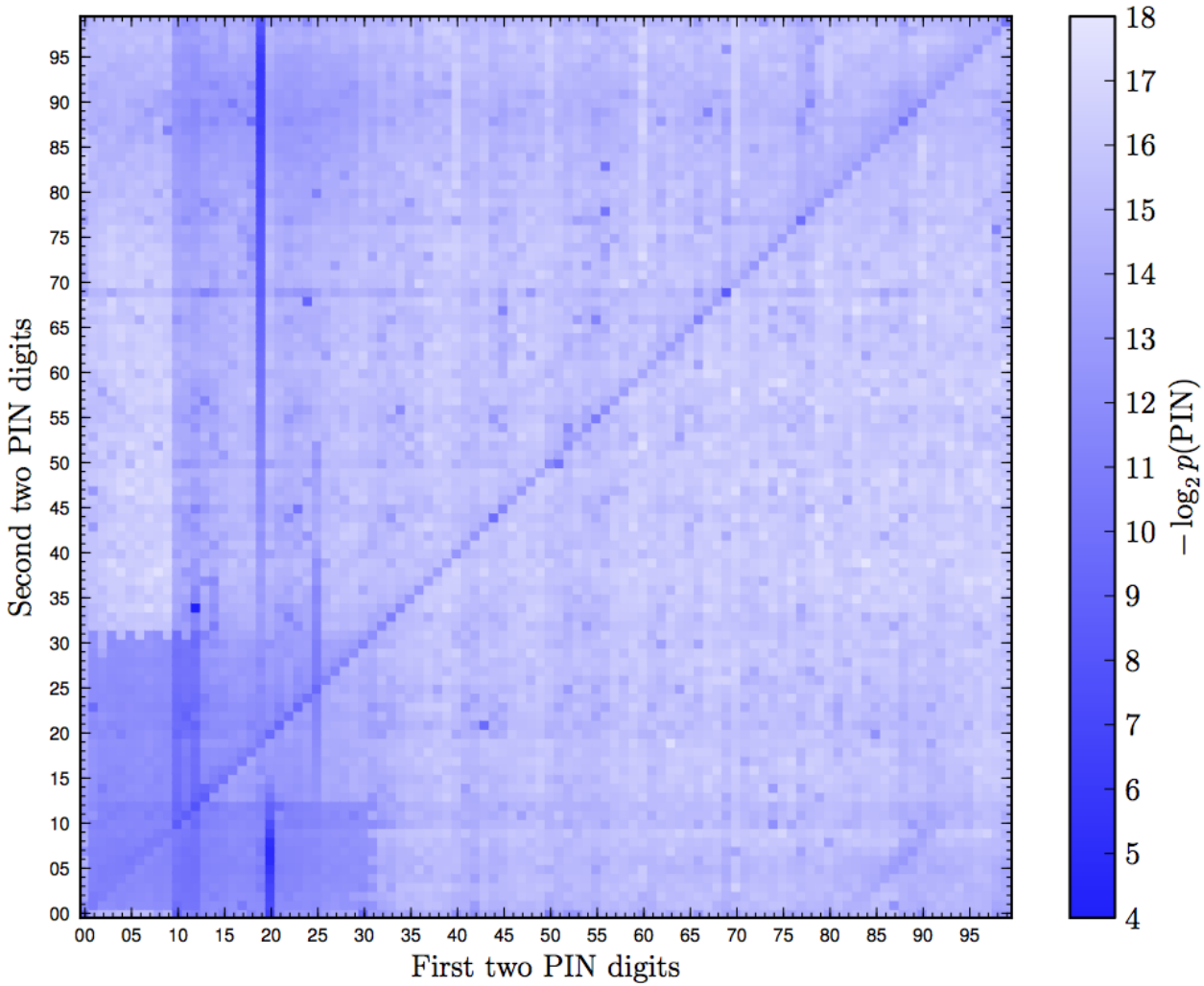
- Jako je teško **motivisati** korisnike da se ponašaju bezbedno u informacionim tehnologijama.
- Sa druge strane, većina stručnjaka u oblasti računarske bezbednosti su matematičari kojima je najvažnije da je njihov protokol matematički neprobojan – a uopšte ne razmatraju upotrebljivost njihovih aplikacija i ljudski element.
- Lozinke su takođe matematički neprobojne, ali samo ukoliko...

Upotrebljivost lozinki

- Lozinke su takođe matematički neprobojne, ali samo ukoliko:
  - Nisu lake za pogoditi;
  - Jesu izdržljive *brute-force* napadima;
  - Jesu (za jednog korisnika) sve različite;
  - Se redovno menjaju;
  - Se lako pamte (ne smeju se zapisivati);
  - ...
- Presek svih ovih zahteva je $\emptyset$!
- Često forsirane lozinke dužine bar n karaktera, koje moraju imati i mala i velika slova i cifre... korisnici ovo zaobilaze na predvidljive načine.
  - "password" $\longrightarrow$ "Password123"...

Ljudski-odabrane tajne

Učestalost četvorocifrenih PIN-ova (passcode) na iPhone aplikaciji:



Jimmy Kimmel Show - What is Your Password?

Bezbednost lozinki

- Osim bezbednosnih problema koji nastaju zbog kršenja nekog od prethodnih zahteva, tu se nalaze i:
  - *Phishing*
  - *Keylogging*
  - Loša implementacija sa serverske strane (spomenuto ranije).
  - ...

Phishing

- Automatizovan socijalni inženjering, najčešće u vidu oponašanja sajtova, u svrhu iznuđivanja privatnih podataka korisnika.
- Prvi put se pojavio kao termin sredinom 90-tih; do 2006. britanske banke su izgubile $\approx$ £35M, a američke $\approx$ \$200M.
- Banke uglavnom olakšavaju posao napadačima:
  - Krše sopstvena pravila, šaljući korisnicima mejlove sa linkovima;
  - Teško razumljive instrukcije za neiskusne korisnike:
    - "Tražite katanac..."
    - "Proverite URL..."

Prepoznavanje *phishing*-a

Koliki procenat ljudi prepoznaje ovaj sajt kao *phishing* sajt?

Prepoznavanje *phishing*-a

- Ovaj sajt očigledno ne koristi zaštićenu konekciju (`http://`) i URL je predugačak i neuobičajen.
- Međutim, **manje od 50%** ispitanih ljudi ga uspešno prepoznaje kao sajt koji nije *PayPal*-ov!
- *Phishing* možda nama deluje kao samo još jedna vrsta neželjene pošte, međutim radi se o ozbiljnom problemu.

Budućnost lozinki

- Lozinke su trenutno popularne prevashodno zato što umnogome olakšavaju posao implementatorima aplikacija:
  - Nema potrebe objašnjavati ih korisnicima;
  - Cena po korisniku zanemarljiva;
  - "Svi ih koriste, što ne bih i ja?".
- Međutim, verovatno neće moći da izdrže eksploziju interneta (i broja sajtova na kojima ćemo morati da imamo naloge).
- *"There is no doubt that over time, people are going to rely less and less on passwords. People use the same password on different systems, they write them down and they just don't meet the challenge for anything you really want to secure."*
  – Bill Gates, 2004.

Pico

- Istraživački projekat predvođen Dr Frank Stajanom iz Kembridžove računarske laboratorije.
- Cilj: ponuditi protokol koji će kompletno zameniti lozinke.
- Zasnovan na fizičkom tokenu (*Pico*) koji komunicira sa aplikacijama umesto korisnika (nekom vrstom dvosmernog izazova) i zaključava se kada nije u okolini dovoljnog broja sitnijih tokena (*"Picosiblings"*).

Pico

Zadaci za vežbu

- Na kraju navedimo dva interesantna zadataka za vežbu; jedan sa teoretskog, drugi sa psihološkog aspekta računarske bezbednosti.

- Oba zadatka su bila zadaci na Kembridžovom završnom ispitu iz računarskih nauka, i predviđeni su da se rade ~ 35 minuta (za studente :)).

Zadaci za vežbu

- Na kraju navedimo dva interesantna zadataka za vežbu; jedan sa teoretskog, drugi sa psihološkog aspekta računarske bezbednosti.

- Oba zadatka su bila zadaci na Kembridžovom završnom ispitu iz računarskih nauka, i predviđeni su da se rade ~ 35 minuta (za studente :)).

Teorijski zadatak

- Dat je protokol kojim server S pomaže osobama A i B da uspostave zajedničku tajnu N_b (koju je izmislio B):

$$\begin{aligned} A \to S: &\quad B, N_a^3 \bmod n \\ S \to B: &\quad A \\ B \to S: &\quad A, N_b^3 \bmod n \\ S \to A: &\quad B, N_a \oplus N_b \end{aligned}$$

  gde je $\oplus$ bitovna ekskluzivna disjunkcija (XOR).

Teorijski zadatak, *cont'd*

- Koja je svrha broja N_a?
- Objasniti kako dva napadača C i D mogu zajedno otkriti tajnu N_b. Pretpostaviti da S nema memoriju.
- Zaustaviti gorenavedeni napad tako što će S čuvati neke podatke u memoriji.
- Objasniti napad kojim C i D i dalje mogu otkriti N_b i pored prethodnih ispravki.
- Zaustaviti gorenavedeni napad popravkom protokola.

Psihološki zadatak

- Ispitna pitanja prolaze kroz navedene pripremne faze pre nego što se prezentuju studentima:
  1. Profesor izmišlja pitanje;
  2. Glavni ispitivač proverava pitanje;
  3. Profesor menja pitanje ukoliko je potrebno;
  4. Eksterni ispitivač proverava pitanje;
  5. Profesor opet menja pitanje ukoliko je potrebno;
  6. Glavni ispitivač odobrava konačnu verziju pitanja;
  7. Službenik štampa pitanje u neophodnom broju kopija.

- Nakon skandala u kome su određeni studenti došli do ispitnih pitanja pre vremena, univerzitet je stavio veliki pritisak na fakultete da se ovo ne ponovi.

Psihološki zadatak, *cont'd*

- Dekan fakulteta na kome se desio skandal je sada jako paranoičan oko računarskih mreža i zahteva da ispitna pitanja ne smeju nipošto biti ni na kojem računaru koji ima pristup mreži sve dok se ispit ne završi.
- Opišite bar četiri načina kojim odlučni student i dalje može doći do ispitnih pitanja pre vremena.[1]
- Opišite bezbednosni protokol kojeg bi trebalo da se drže svi članovi fakulteta, uzimajući u obzir potrebe dekana i faze pripreme ispitnih pitanja.

[1] Hint: legitiman odgovor je i "Odvesti profesora u pab, napiti ga, i onda mu postavljati pitanja".