Kriptografija i kriptoanaliza

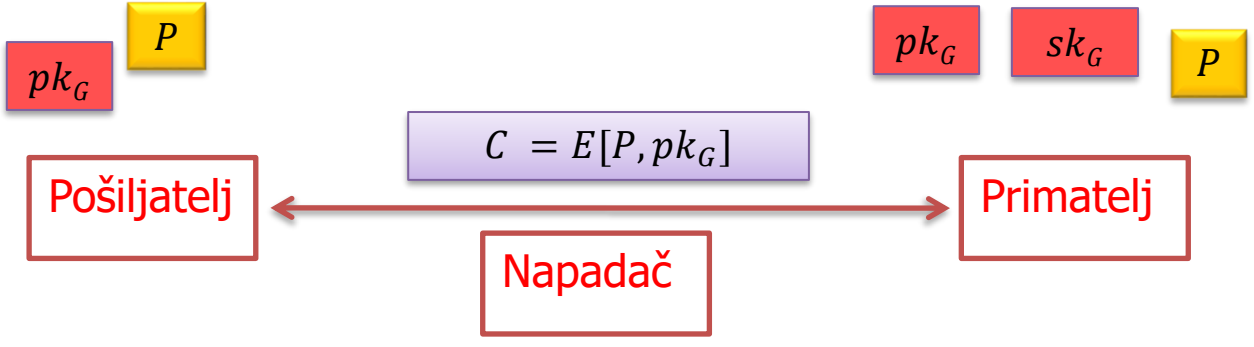
doc. dr. sc. Ante Đerek i prof. dr. sc. Marin Golub

6.
Asimetrični kriptosustavi

Napadi na kriptosustav RSA

Ponavljanje: Enkripcija javnim ključem

- Nova ideja: Primatelj ima dva ključa
  - Javni ključ pk_G: Javno poznat (npr. telefonski imenik)
  - Privatni ključ sk_G: Poznat samo Primatelju
  - Jasni tekst se kriptira s javnim ključem
  - Skriveni tekst se dekriptira s privatnim ključem

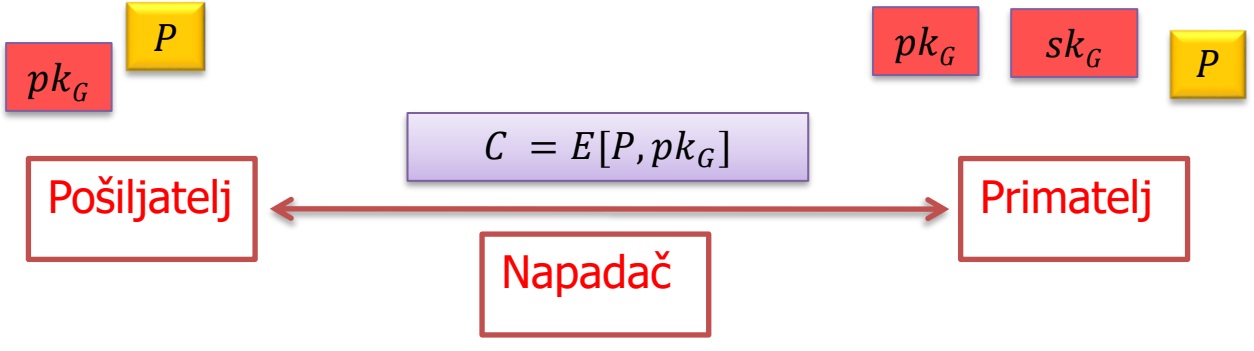


Ponavljanje: Sustav kriptiranja javnim ključem

- Trojka *efikasnih* algoritama G, E i D
  - G – algoritam koji generira par ključeva pk, sk
  - $E(m, pk)$ – algoritam enkripcije
  - $D(c, sk)$ – algoritam dekripcije
- Za svaki par ključeva (pk, sk) generiranih algoritmom G i za svaki jasni tekst p vrijedi $D(E(p, pk), sk) = p$

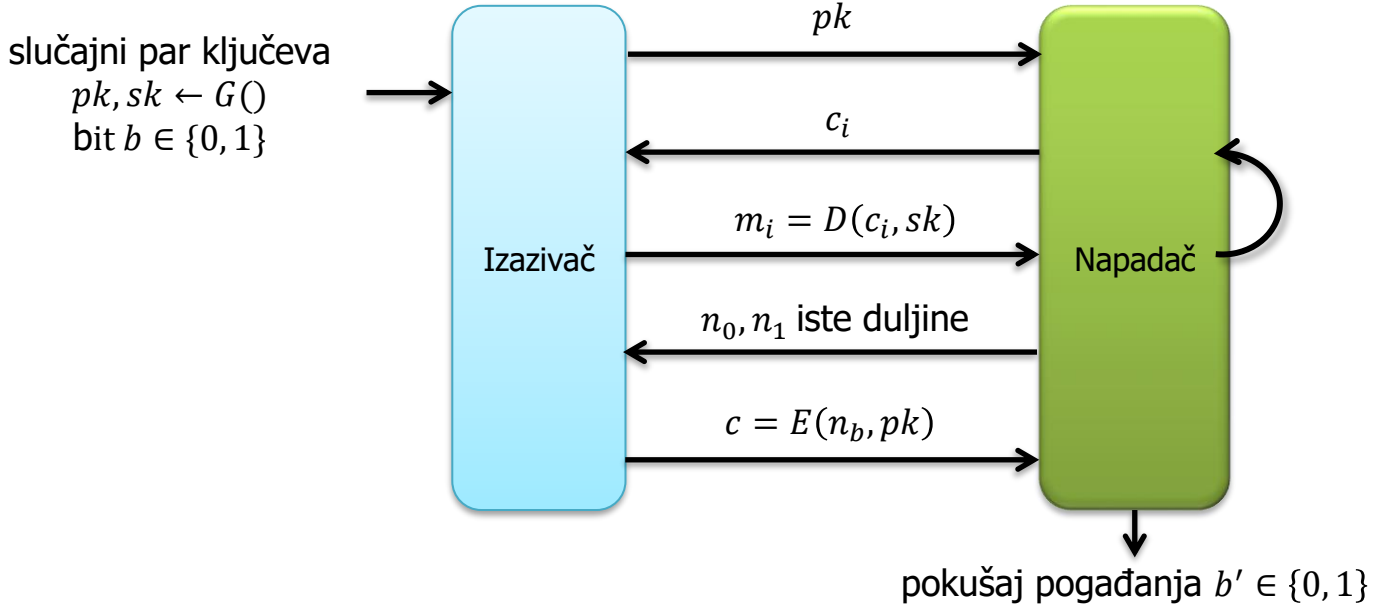
Sustav kriptiranja javnim ključem – sigurnost

- SKJK je *siguran* ako je teško na temelju kriptiranog teksta odrediti bilo što o jasnom tekstu ...
- ... čak i ako napadač ima na raspolaganju:
  - Javni ključ kojim je jasni tekst kriptiran
    - (chosen-plaintext attack).
  - Mogućnost da dobije $p = D(c, sk)$ za proizvoljni c
    - (chosen-ciphertext attack)



Primjer definicije sigurnosti SKJK

- Semantička sigurnost od napada odabranim skrivenim tekstom (semantic security under chosen-ciphertext attack): Svaki efikasan algoritam ima zanemarivu prednost u sljedećoj igri.



$$\mathrm{Adv}_{SS-CCA1}(A) = |P(W_0) - P(W_1)|$$

Ponavljanje: Obični RSA

Algoritam G:

1. Veliki slučajni prosti brojeve p i q
2. Izračunam $N = p \cdot q$
3. Izračunam $\varphi(N) = (p-1)(q-1)$
4. Odaberem $e \in \mathbb{Z}^*_{\varphi(N)}$
5. Izračunam $d = e^{-1}$ u $\mathbb{Z}^*_{\varphi(N)}$
6. Javni ključ: $pk = (e, N)$
7. Privatni ključ: $sk = (d, N)$

Algoritam E:

- $E\big(m, (e, N)\big) = m^e$ u $\mathbb{Z}_N$

Algoritam D:

- $D\big(c, (d, N)\big) = c^d$ u $\mathbb{Z}_N$

Zadatak: Obični RSA 4

- Šaljemo 64-bitni DES ključ K koristeći RSA
  - neka je $e = 65537$
  - šaljemo $c = RSA(K, pk)$

- Ponekad će se slučajno dogoditi da je $K = K_1 \cdot K_2$ gdje su K_1 i K_2 32-bitni brojevi
- $c = K^e = K_1^e \cdot K_2^e$ u $\mathbb{Z}_N$
- $c \cdot (K_1^e)^{-1} = K_2^e$ u $\mathbb{Z}_N$
- *Meet-in-the-middle* algoritam nalazi K_1 i K_2 u 2^{32} koraka

RSA – Kombinacija sa simetričnom enkripcijom

- U praksi se RSA gotovo nikad ne koristi za kriptiranje podataka već kriptiranje materijala za ključ.
- Puno konstrukcija.
  - Kriptiranje materijala za ključ
  - Digitalna omotnica

RSA – Kombinacija sa simetričnom enkripcijom

- Dokazivo ispravna konstrukcija, pod određenim jakim pretpostavkama na sigurnost običnog RSA, hash funkcija i simetrične enkripcije.

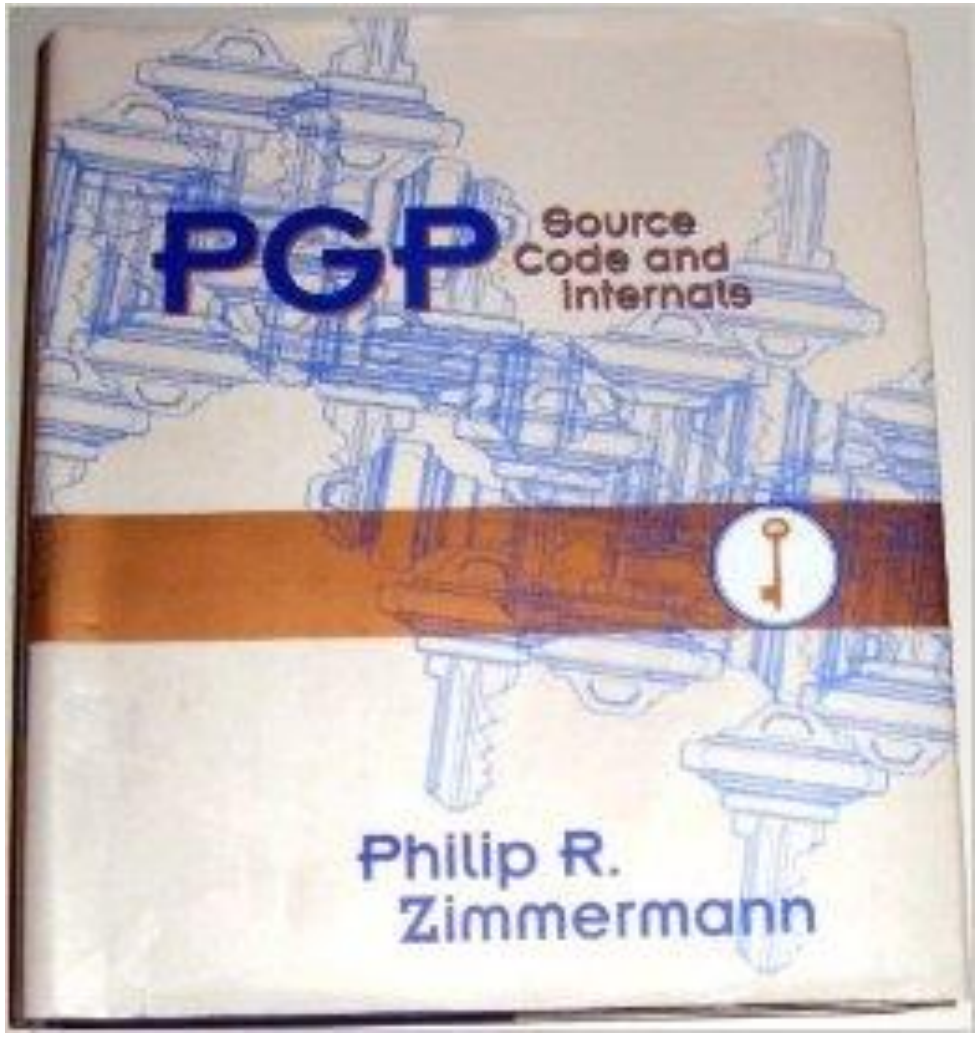
H je hash funkcija, E_s simetrična enkripcija

Algoritam E:

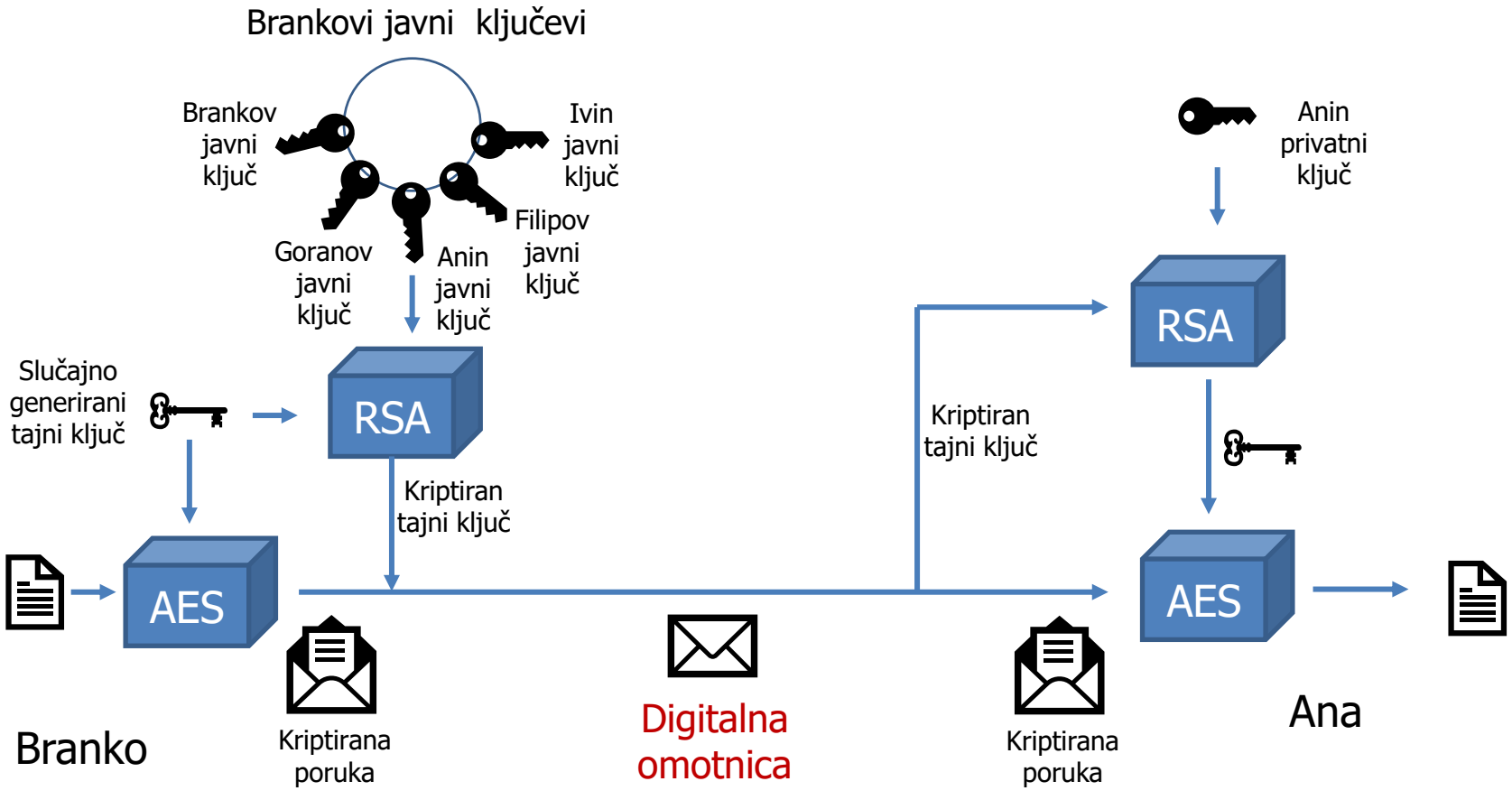
1. Izaberem slučajni $x \in \mathbb{Z}_N$
2. Izračunam $k = H(x)$
3. Izračunam $c_1 = E(x, pk)$
4. Izračunam $c_2 = E_s(m, k)$
5. Skriveni tekst je (c_1, c_2)

RSA – Digitalna omotnica

- Generiramo novi nasumični ključ K
  - Poruku kriptiramo ključem K pomoću simetrične enkripcije.
  - Ključ K kriptiramo sustavom RSA
- $E(Pad(K), pk), E_S(m, K)$
- Primijetite da je digitalna omotnica nije deterministička enkripcija!


PGP
Source Code and Internals
Philip R. Zimmermann

Kako osigurati tajnost?



RSA – *Padding*

- Jasni tekst se uvijek nadopunjuje na zadanu veličinu!
- Postupak nadopunjavanja (*padding*) igra kritičnu ulogu i pažljivo je osmišljen.
  - PKCS#1 v1.5 (mnoštvo sigurnosnih problema)
  - OAEP

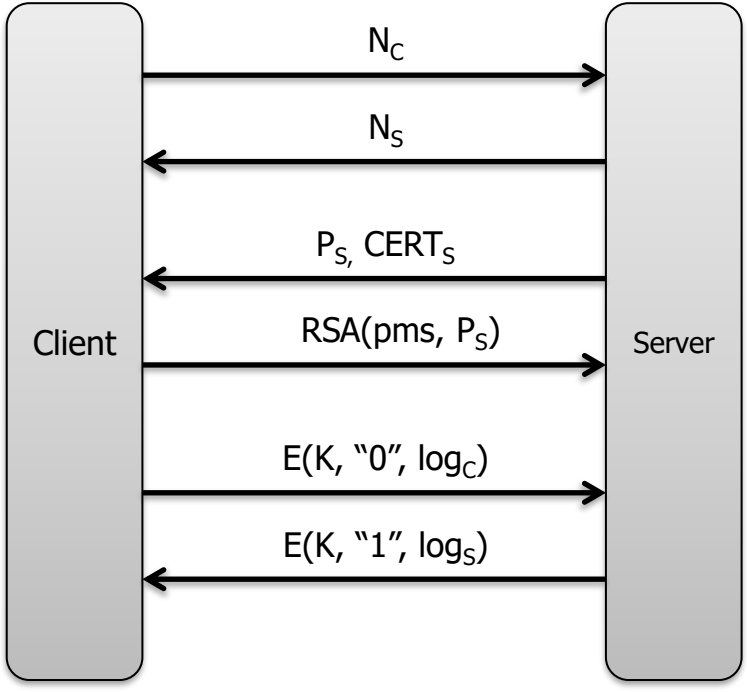
RSA – PKCS#1 v1.5 *Padding*

```
EME-PKCS1-v1_5 encoding:

a.  Generate an octet string PS of length k - mLen - 3
    consisting of pseudo-randomly generated nonzero octets.
    The length of PS will be at least eight octets.

b.  Concatenate PS, the message M, and other padding to form
    an encoded message EM of length k octets as

       EM = 0x00 || 0x02 || PS || 0x00 || M.
```

Izvor: ietf.org

Bleichenbacherov napad

- Pretpostavimo da TLS poslužitelj koristi PKCS#1 v1.5 *padding*
- Razumna implementacija:
  - Izračunaj $m = c^d \text{ u } \mathbb{Z}_N$
  - Ako m ne počinje s bajtovima 0x00 0x02 vrati klijentu poruku „Bad padding”.
- Dobili smo oracle!
  - Napadač za proizvoljni broj c može saznati da li $c^d \text{ u } \mathbb{Z}_N$ počinje s 0x00 0x02.



Bleichenbacherov napad

- Dobili smo oracle!
  - Napadač za proizvoljni broj c može saznati da li c^d u $\mathbb{Z}_N$ počinje s 0x00 0x02, gdje je (d, N) privatni ključ.
- Oracle se može iskoristiti – postoji efikasan algoritam koji nakon razumnog broja upita određuje privatni ključ.
  - Oko 2M upita potrebno za 1024-bitni RSA ključ.
- *Daniel Bleichenbacher, Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1 (1998)*

Bleichenbacherov napad – pouke?

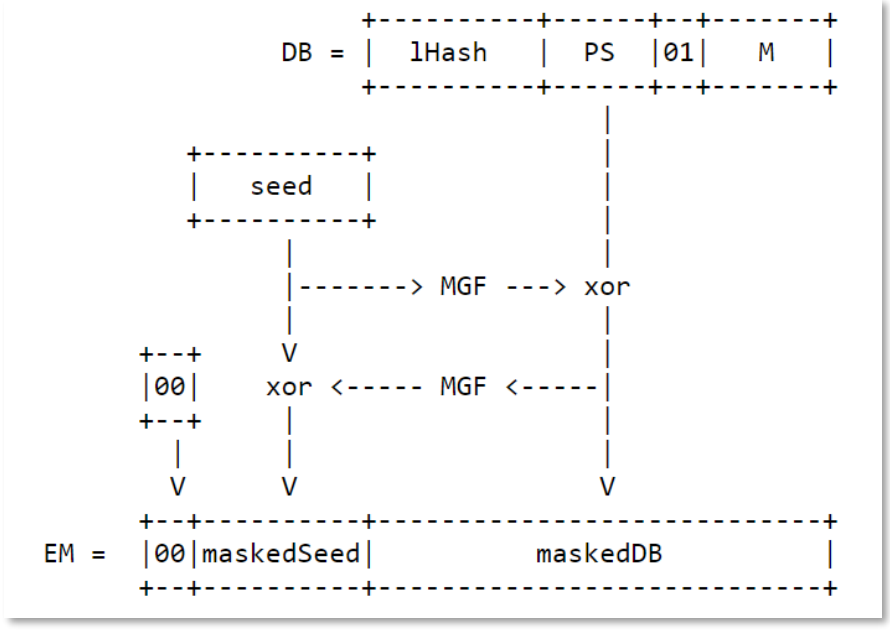
- Implementacija ne smije napadaču (tj. Korisniku) otkrivati detalje greške.
- Napad odabranim skrivenim tekstom je stvarno razumna definicija sigurnosti.

> In any case, a TLS server MUST NOT generate an alert if processing an RSA-encrypted premaster secret message fails, or the version number is not as expected. Instead, it MUST continue the handshake with a randomly generated premaster secret. It may be useful to log the real cause of failure for troubleshooting purposes; however, care must be taken to avoid leaking the information to an attacker (through, e.g., timing, log files, or other channels.)

Izvor: https://datatracker.ietf.org/doc/html/rfc5246

RSA – OAEP *Padding*

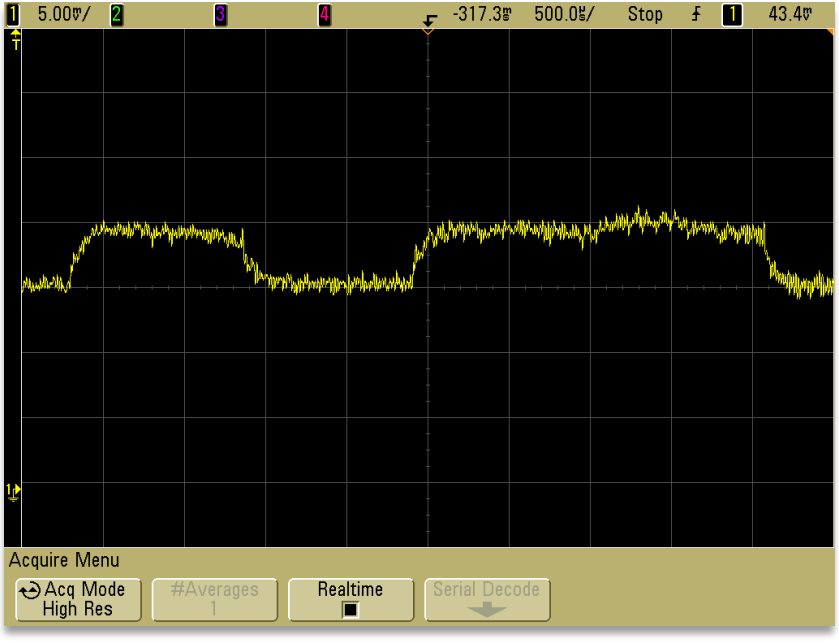
- *Optimal asymmetric encryption padding*
  - Struktura slična Feistelovoj mreži
- Dokazano sigurna (u smislu semantičke sigurnosti protiv napada odabranim skrivenim tekstom) pod jakim pretpostavkama sigurnosti običnog RSA i hash funkcija.
  - Osnovna ideja: *„all-or-nothing security"* – ako napadač ne sazna sve bitove od EM onda ne može saznati ništa o m.



Izvor: ietf.org

Side channel napadi na RSA

```
// Modularno eksponenciranje
// b^a mod n, a u binarnom zapisu

d = 1;
i = m;
dok je (i >= 0) {
    d = (d * d) mod n;
    ako je (a[i] == 1) {
        d = (d*b) mod n;
    }
    i --;
}
```



Izvor:
wikipedia.org

Napadi na RSA

- Matematički napadi:
  - Wienerov napad na mali privatni eksponent.
  - Franklinov and Reiterov napad na povezane poruke.
  - …
- Implementacijski napadi:
  - Napadi mjerenjem vremena.
  - Napadi u slučaju sitnih grešaka pri računanju.
  - …
- Izvor: *Dan Boneh, Twenty years of attacks on the RSA cryptosystem*

RSA – sigurnost

- Obični RSA je nesiguran!
- Ako se RSA ispravno koristi smatramo ga sigurnim!
- Puno implementacijskih napada!
- Najbolji poznati općeniti napad:
  - Faktorizacija modula
  - Na primjer, algoritmom GNFS (General Number Field Sieve)
  - U 2020. najveći faktorizirani modul je veličine 829 bitova.

RSA – faktorizacija modula

paul zimmermann Paul.Zimmermann at inria.fr
Fri Feb 28 16:48:03 CET 2020

- Previous message: [Cado-nfs-discuss] move to gitlab
- **Messages sorted by:** [date] [thread] [subject] [author]

```
Date: February 28, 2020

For the past three months, ever since the DLP-240 record announced in
December 2019 [1], we have been in a historically unique state of
affairs: the discrete logarithm record (in a prime field) has been
larger than the integer factorization record. We are pleased to
rectify this situation with the factorization of RSA-250 from the RSA
challenge list:

RSA-250 =
214032465024074496126442307283933356300861471514475501779775492088141802344714013664334
        =
641352894770715802787901901705773890848250147429434472081168596320245323446302386235987
        *
333720275949781565562260106053551142279407603447675546667845209870238417292100370802574

This computation was performed with the Number Field Sieve algorithm,
using the open-source CADO-NFS software [2].

The total computation time was roughly 2700 core-years, using Intel Xeon
Gold 6130 CPUs as a reference (2.1GHz):

    RSA-250 sieving:  2450 physical core-years
    RSA-250 matrix:    250 physical core-years
```

RSA – faktorizacija modula

paul zimmermann Paul.Zimmermann at inria.fr
Fri Feb 28 16:48:03 CET 2020

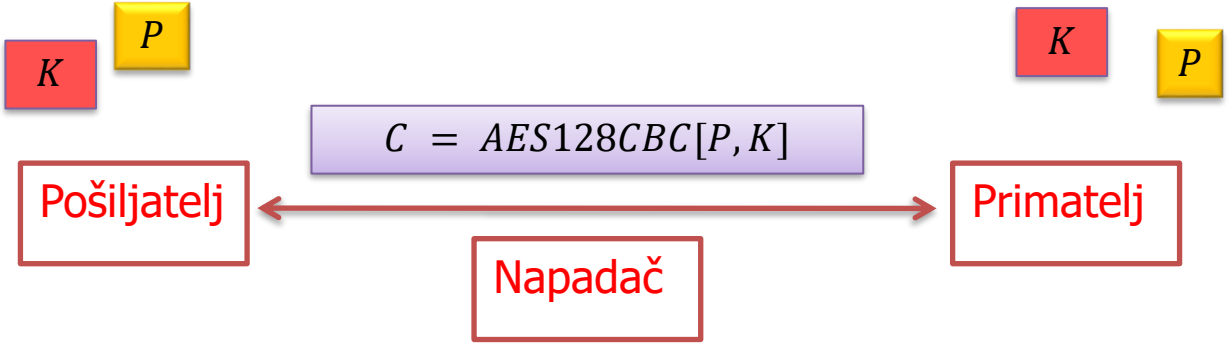
- Previous message: [Cado-nfs-discuss] move to gitlab
- **Messages sorted by:** [date] [thread] [subject] [author]

```
Date: February 28, 2020

For the past three months, ever since the DLP-240 record announced in
December 2019 [1], we have been in a historically unique state of
affairs: the discrete logarithm record (in a prime field) has been
larger than the integer factorization record. We are pleased to
rectify this situation with the factorization of RSA-250 from the RSA
challenge list:

RSA-250 =
214032465024074496126442307283933356300861471514475501779775492088141802344714013664334
        =
641352894770715802787901901705773890848250147429434472081168596320245323446302386235987
        *
333720275949781565562260106053551142279407603447675546667845209870238417292100370802574

This computation was performed with the Number Field Sieve algorithm,
using the open-source CADO-NFS software [2].

The total computation time was roughly 2700 core-years, using Intel Xeon
Gold 6130 CPUs as a reference (2.1GHz):

    RSA-250 sieving:  2450 physical core-years
    RSA-250 matrix:    250 physical core-years
```

6.
Asimetrični kriptosustavi

Digitalni potpis zasnovan na RSA

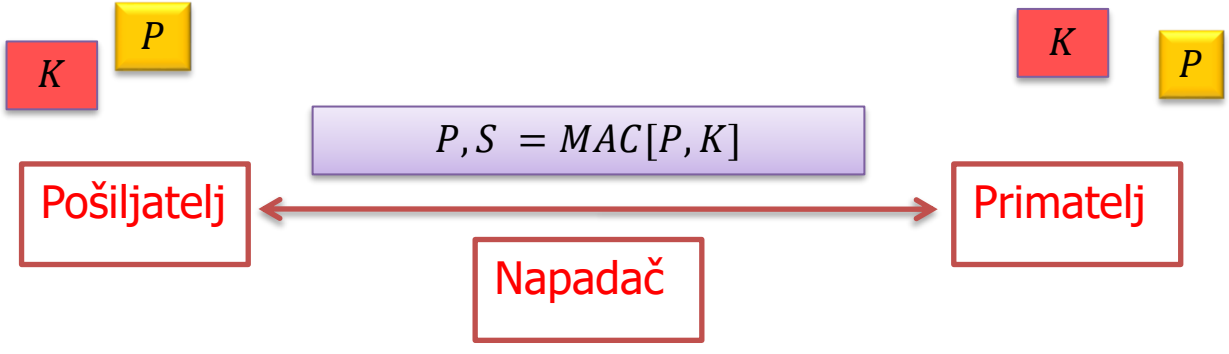
Enkripcija ne rješava sve probleme!



- Ako ste primili i uspješno dekriptirali poruku možete li biti sigurni da znate:
  - Tko je generirao poruku?
  - Je li dekriptirana poruka identična originalnoj?

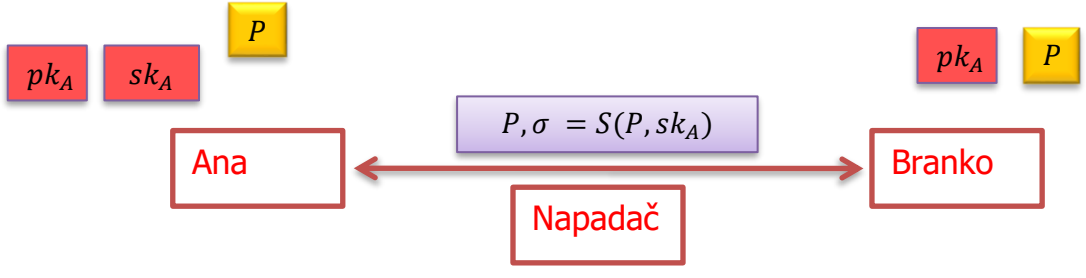
MAC / Autentificirana enkripcija

- Kod za integritet poruke (*Message Authentication Code*)
- Autentificirana enkripcija

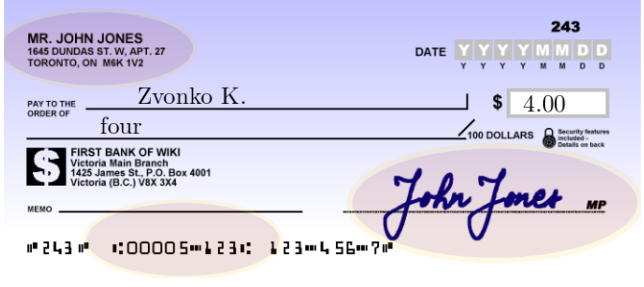


Javni i tajni ključevi

- Stara ideja: Svatko ima dva ključa
  - Javni ključ pk_A: Javno poznat (npr. telefonski imenik)
  - Privatni ključ sk_A: Poznat samo Ani
  - Ana *generira* potpis svojim privatnim ključem sk_A
  - Branko *provjerava* potpis Aninim javnim ključem pk_A



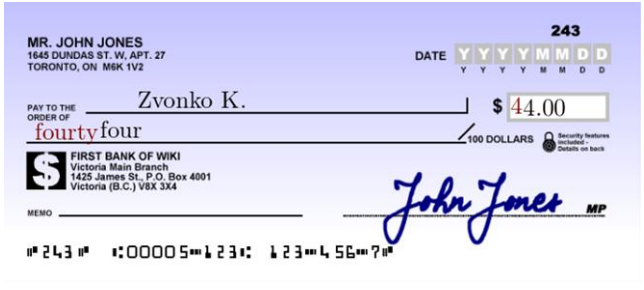
Digitalni vs analogni potpis – autentičnost



Izvor: wikipedia.org

- Svatko može provjeriti ispravnost digitalnog potpisa ako ima na raspolaganju javni ključ tobožnjeg potpisnika.
- Provjera ispravnosti je garancija da je potpis stvarno generiran odgovarajućim privatnim ključem.
- Veza između ključeva i identiteta?

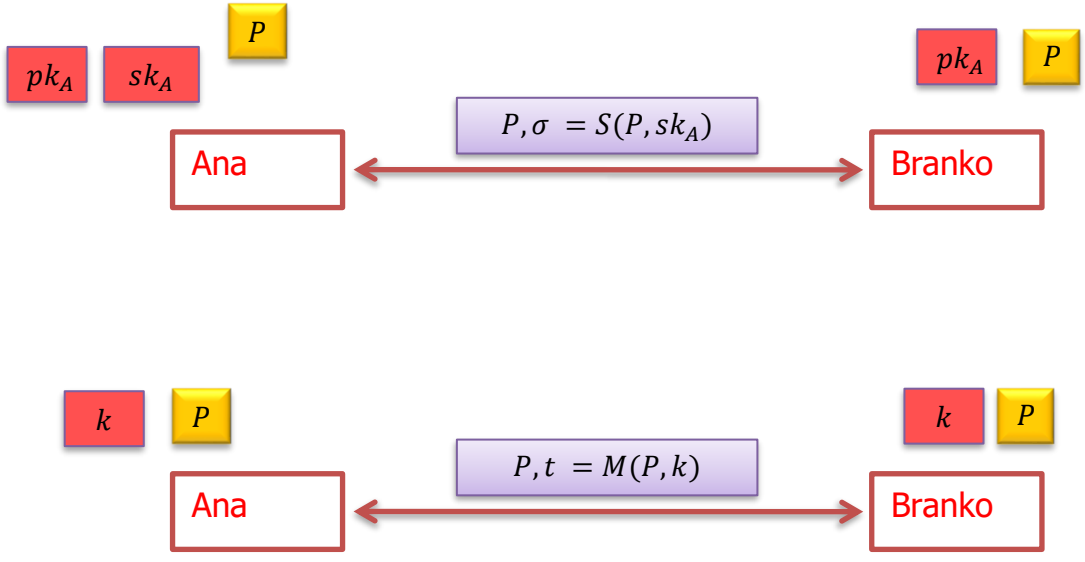
Digitalni vs analogni potpis – integritet



Izvor: wikipedia.org

- Digitalni potpis je vezan uz dokument.
- Ispravan potpis garantira integritet dokumenta.

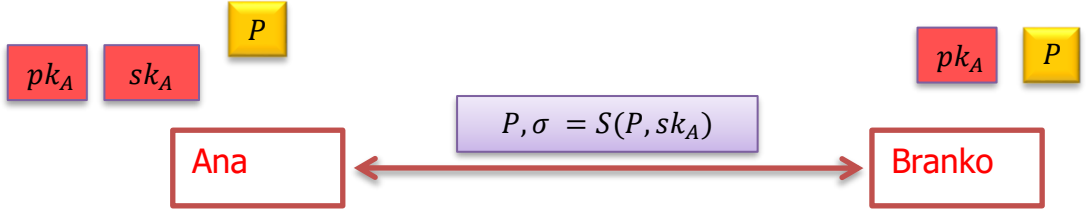
Digitalni potpis vs MAC – neporecivost (non-repudiation)



- Moguće je trećoj strani dokazati da je pošiljatelj potpisao poruku!
- Veza između ključeva i identiteta?
- „Netko me je hakirao” obrana?

Sustav digitalnog potpisa

- Trojka efikasnih algoritama G, S i V
  - G – algoritam koji generira par ključeva pk, sk
  - $S(m, sk)$ – algoritam potpisivanja
  - $V(m, \sigma, pk)$ – algoritam verifikacije
- Za svaki par ključeva (pk, sk) generiranih algoritmom G i za svaki jasni tekst p vrijedi $V(p, S(p, sk), pk) = 1$

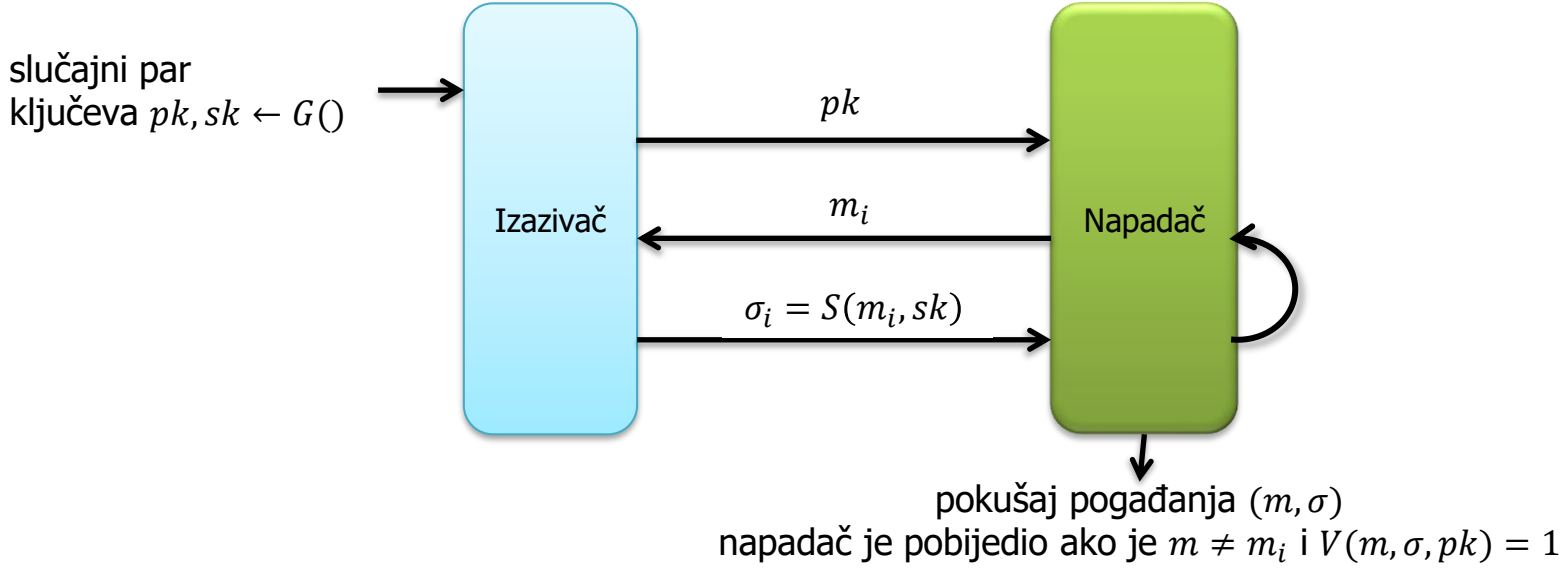


Sustav digitalnog potpisa – sigurnost

- SDP je siguran ako je teško odrediti bilo koju poruku p i bilo koji potpis (niz bitova) σ takav da
  - $V(p, \sigma, pk) = 1$
  - p nikad nije potpisan s privatnim ključem sk
- ... čak i ako napadač ima na raspolaganju:
  - Javni ključ pk
  - Mogućnost da dobije potpis $S(p, sk)$ za proizvoljnu poruku p (chosen message attack)

Primjer definicije sigurnosti digitalnog potpisa

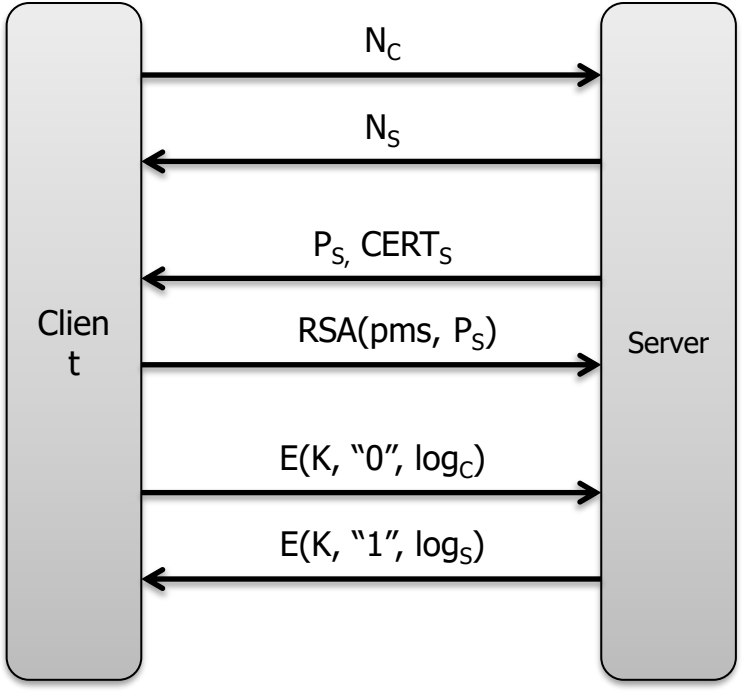
Nemogućnost krivotvorenja potpisa bilo kakve poruke pod napadom odabranom porukom (*existential unforgeability under chosen message attack):* Mti jedan algoritam koji koristi razumne resurse ne može pobijediti u sljedećoj igri s vjerojatnošću nezanemarivo većom od nule.

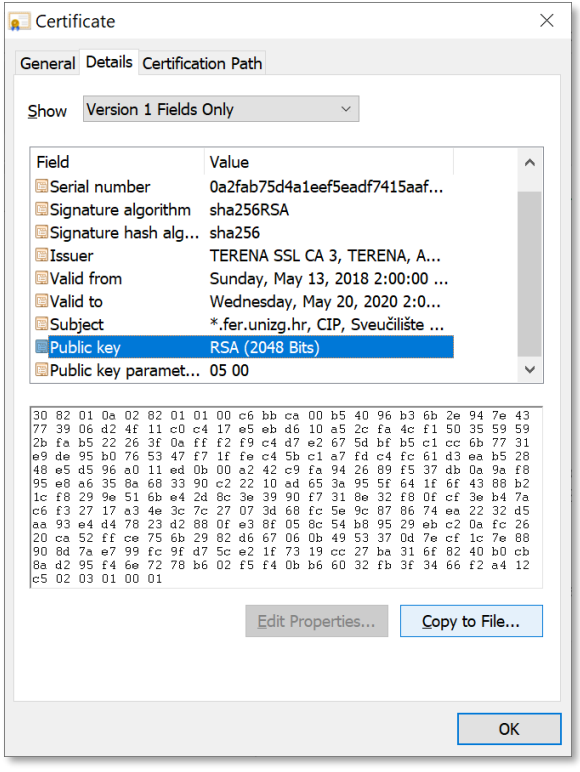


Digitalni potpis – primjene

- Potpisivanje digitalnih dokumenata
- Sigurnosni protokoli (TLS, …)
- Autentifikacija email-a
- Provjera autentičnosti softvera (apk, exe, firmware, …)
- Kriptovalute
- …

Primjena – TLS protokol

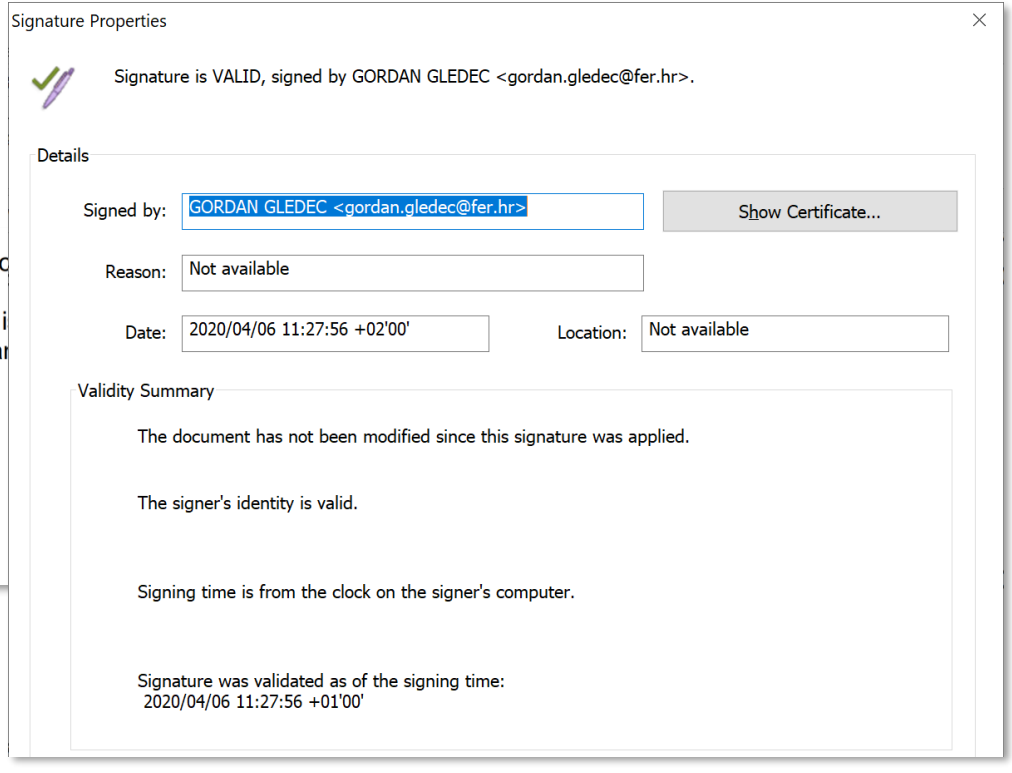




Primjena – e-Dokumenti

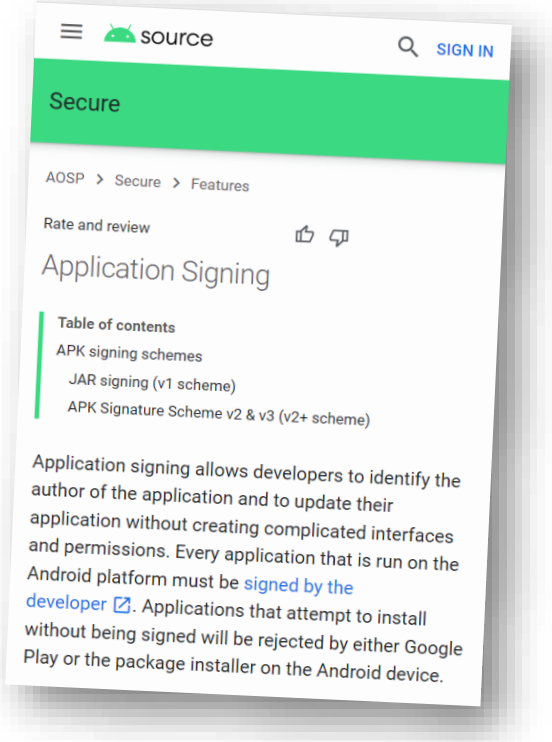
- ovisno o razvoju situacije, razmotrit će se uvo

II. Ova odluka je privremenog karaktera, donosi se i
okolnosti navedenih u točki I., stupa na snagu dan



Primjena – Android mobilne aplikacije

- Svaka mobilna aplikacija mora biti digitalno potpisana od strane autora!
- Operacijski sustav ne dopušta instaliranje i pokretanje nepotpisane aplikacije.
- Aplikacija može biti potpisana *bilo kojim* ključem.
  - Ključ je dio paketa koji sadrži aplikaciju i potpis.
- Aplikacije potpisane istim ključem mogu dijeliti podatke.



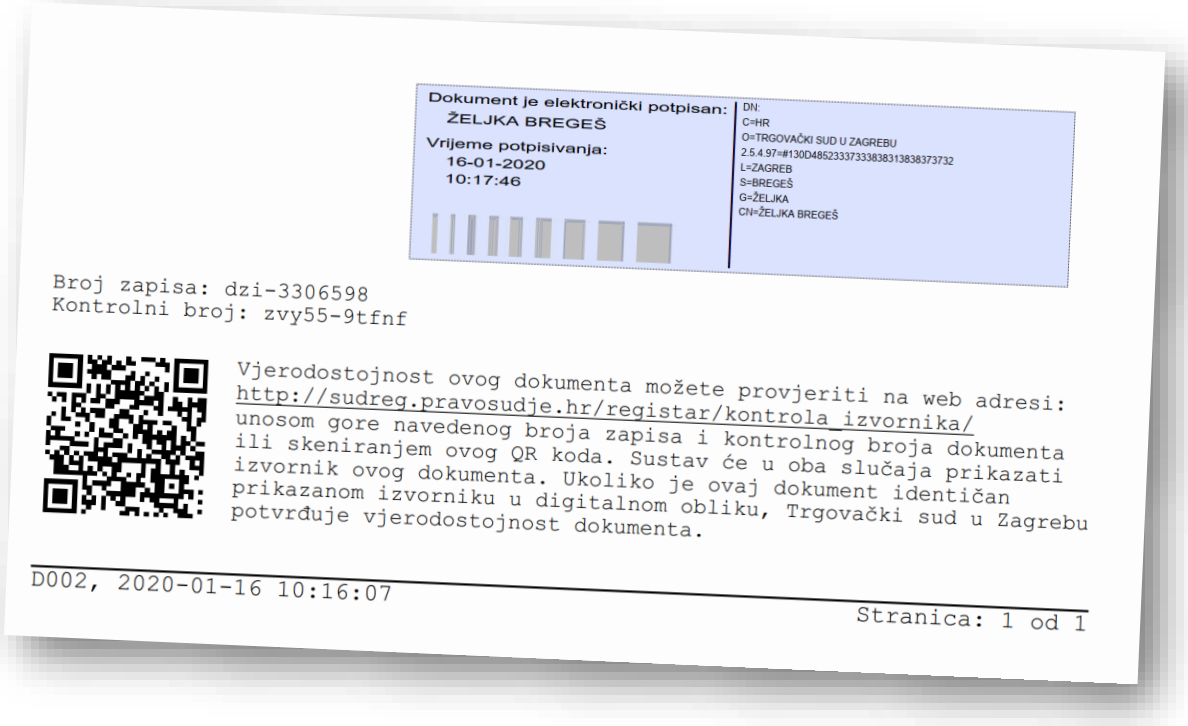
Izvor: source.android.com

Primjena – COVID potvrde

Izvor: Interoperability of health certificates Trust framework

Što sve *nije* digitalni potpis?

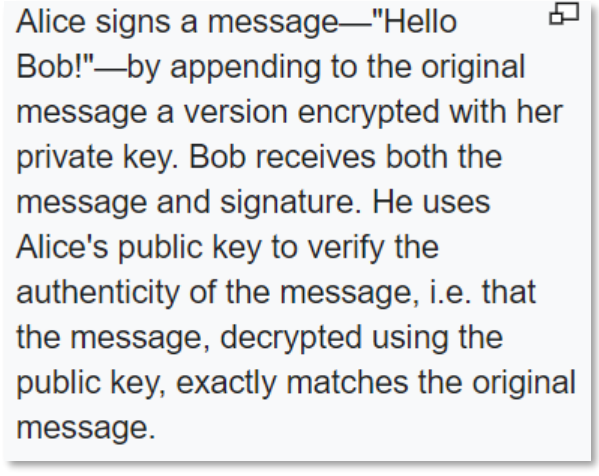
- Tekst koji kaže da je dokument digitalno potpisan.
- Broj koji omogućuje dohvaćanje originalnog dokumenta online.
- QR kod.
- Slika analognog potpisa.
- …



Primjeri sustava digitalnog potpisa

- RSA (1978)
  - teorija brojeva, sigurnost povezana s problemom faktorizacije
- McEliece (1978)
  - teorija kodiranja, sigurnost povezana s problemom dekodiranja općenitog linearnog koda
- ElGamal (1985)
  - teorija brojeva ili eliptičke krivulje, sigurnost povezana s problemom diskretnog logaritma
- Schnorr (1991)
  - Jednostavan i efikasan sustav, sigurnost povezana s problemom diskretnog logaritma
- DSA (1992)
  - vrlo slično ElGamalovim potpisima

Digitalni potpisi i asimetrične šifre



Izvor: https://en.wikipedia.org/wiki/Digital_signature (ožujak 2021.)

- Digitalni potpis nije enkripcija sažetka poruke privatnim ključem!
- Često (ali ne i uvijek) se ista matematička ideja može iskoristiti za izgradnju asimetrične šifre i digitalnog potpisa.
  - RSA šifra i RSA potpis
  - Diffie-Hellman: ElGamal šifra, DSA potpis

„Obični RSA” digitalni potpis

Algoritam S:

- $S\big(m,(d,N)\big) = m^d \text{ u } \mathbb{Z}_N$

Algoritam V:

- $V\big(m,\sigma,(e,N)\big) = (\sigma^e == m \text{ u } \mathbb{Z}_N)\ ?\ 1 : 0$

Zadatak: Obični RSA potpis 1

- Može li napadač na temelju javnog ključa (e, N) pronaći bilo koju poruku i njen ispravan potpis?

- Odaberem proizvoljni $x \in \mathbb{Z}_N$
- Izračunam $y = x^e$ u $\mathbb{Z}_N$
- x je ispravan potpis za poruku y.

Zadatak: Obični RSA potpis 2

- Pretpostavimo da napadač ima dvije poruke i njihove ispravne potpise, može li ih kombinirati tako da dobije ispravan potpis za neku novu poruku?

- $m_1, \sigma_1 = m_1^d$ u $\mathbb{Z}_N$
- $m_2, \sigma_2 = m_2^d$ u $\mathbb{Z}_N$
- $\sigma_1 \cdot \sigma_2 = m_1^d \cdot m_2^d = (m_1 \cdot m_2)^d$ u $\mathbb{Z}_N$
- $\sigma_1 \cdot \sigma_2$ je ispravan potpis od $m_1 \cdot m_2$

Zadatak: Obični RSA potpis 3

- Napadač ima mogućnost dobiti potpis za točno jednu poruku koja izgleda slučajno. Želi iskoristiti tu mogućnost kako bi dobio potpis konkretne poruke m po njegovom izboru.

RSA digitalni potpis

H – kriptografska funkcija sažetka
Pad – funkcija nadopunjavanja

Algoritam S:

- $S\big(m,(d,N)\big) = Pad(H(m))^d \text{ u } \mathbb{Z}_N$

Algoritam V:

- $V\big(m,\sigma,(e,N)\big) = (Unpad(\sigma^e \text{ u } \mathbb{Z}_N) == H(m)) \; ? \; 1 : 0$

Zadatak: Obični RSA potpis 3

- Zašto isti napad više ne radi?

RSA digitalni potpis – Padding

- Hash poruke se uvijek nadopunjuje na zadanu veličinu!
- Postupak nadopunjavanja (*padding*) igra kritičnu ulogu i pažljivo je osmišljen.
  - PKCS#1 v1.5 (mnoštvo sigurnosnih problema)
  - PSS

RSA – PKCS#1 v1.5 Padding

4. Generate an octet string PS consisting of emLen - tLen - 3 octets with hexadecimal value 0xff. The length of PS will be at least 8 octets.

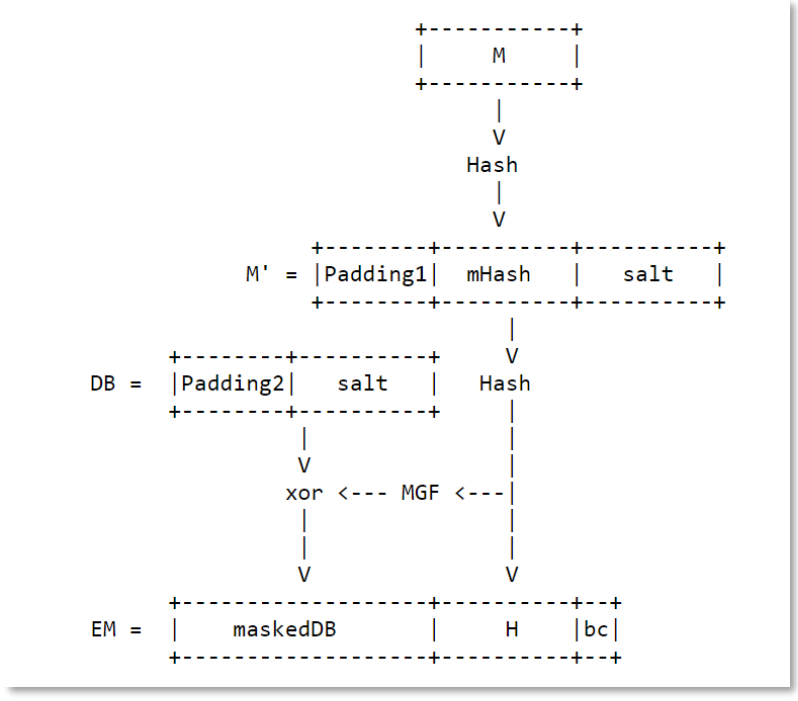
5. Concatenate PS, the DER encoding T, and other padding to form the encoded message EM as

   EM = 0x00 || 0x01 || PS || 0x00 || T.

Izvor: https://datatracker.ietf.org/doc/html/rfc8017

RSA – *PSS Padding*

- *Probabilistic signature scheme*
- Dokazano sigurna pod jakim pretpostavkama sigurnosti običnog RSA i hash funkcija.
- *Mihir Bellare , Phillip Rogaway, PSS: Provably Secure Encoding Method for Digital Signatures (1998)*



Izvor: https://datatracker.ietf.org/doc/html/rfc8017