## Sigurnost računalnih sustava

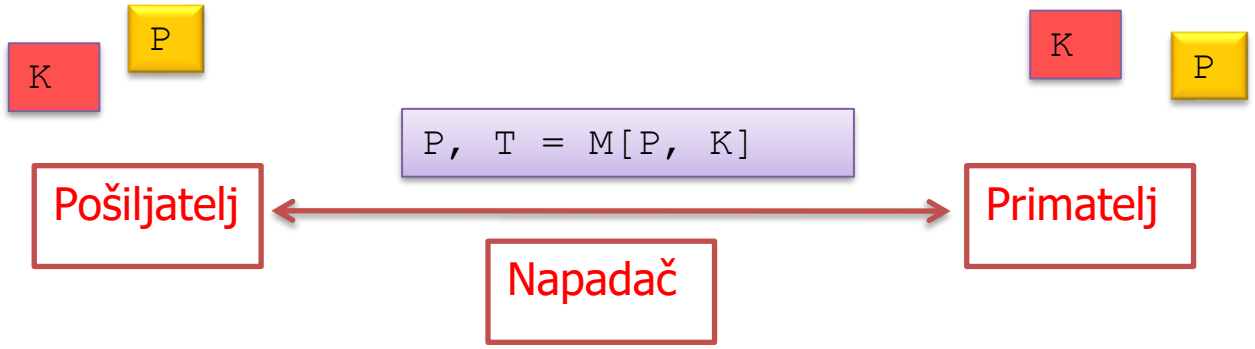
# Osnove kriptografije i kriptoanalize

doc. dr. sc. Ante Đerek
doc. dr. sc. Stjepan Groš
izv. prof. dr. sc. Miljenko Mikuc
izv. prof. dr. sc. Marin Vuković

Osnove kriptografije i kriptoanalize

# Kodovi za integritet poruke

# Kako osigurati integritet komunikacije?



# Kod za integritet poruke

$M$ je deterministički algoritam $M: \{0,1\}^* \times \{0,1\}^k \rightarrow \{0,1\}^n$
koji proizvoljnoj *poruci* i *ključu* pridružuje *oznaku (eng. tag)* fiksne duljine.

*Message Authentication Code (MAC)* ili *Message Integrity Code (MIC)*

# Sigurnost koda za integritet poruke – neformalno

- Kod za integritet poruke je siguran ako je vrlo teško krivotvoriti oznaku, odnosno generirati ispravnu oznaku za proizvoljnu poruku.
- … čak i ako napadač ima na raspolaganju mnogo parova $(m_i, t_i)$ gdje je $t_i = M(m_i, k)$.

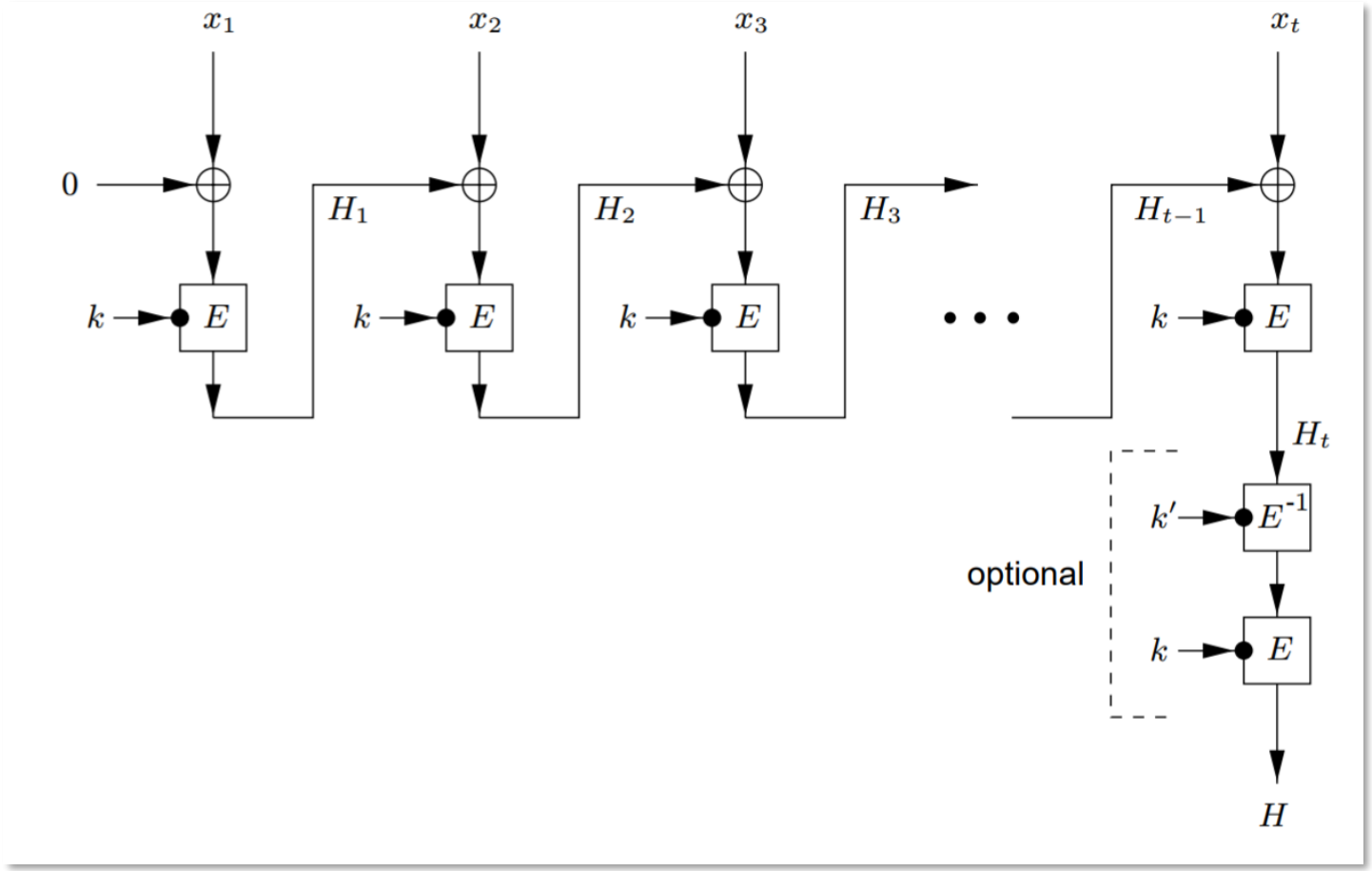
# HMAC – MAC pomoću hash funkcija

- Primjena kriptografskih hash funkcija
- Definiran u RFC2104

$$HMAC(m,k) = H(k \oplus opad \,||H(k \oplus ipad||m))$$

# MAC pomoću blok šifre

- Primjer: CBC-MAC




Izvor: Handbook of Applied Cryptography

# Primjeri kodova za integritet poruka

- HMAC konstrukcija
  - bazirana na kriptografskim funkcijama sažetka
- CBC-MAC, OMAC, PMAC konstrukcije
  - bazirani na blok šiframa
- Poly1305 (2005.)
  - baziran na univerzalnim funkcijama sažetka (*unversal hashing) i blok šiframa*

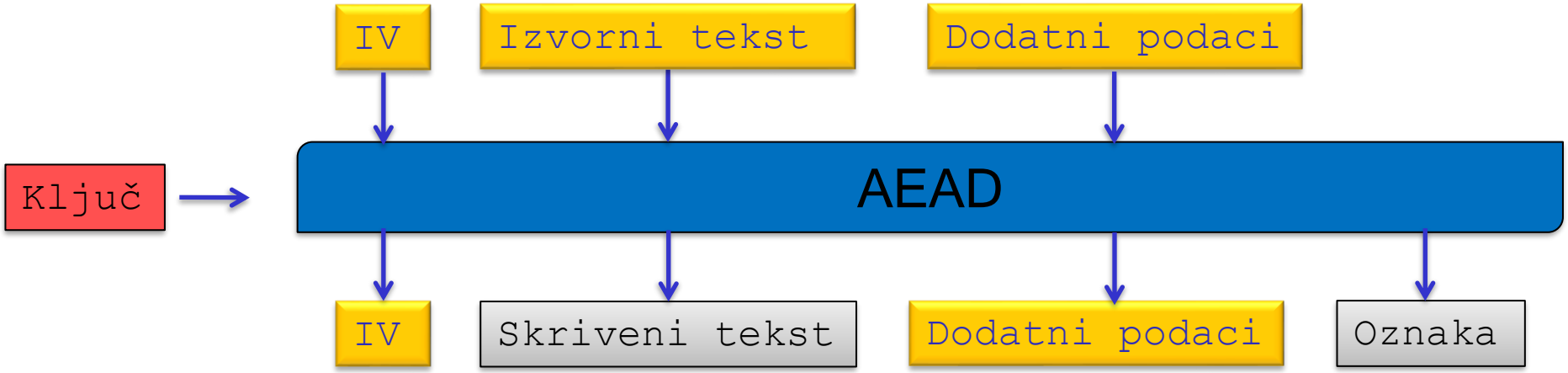
# Povjerljivost i integritet?

- Encrypt-and-MAC: $E(m, k_1), M(m, k_2)$
  - SSH, generalno nesigurna konstrukcija
- MAC-then-Encrypt: $E(m \,||M(m, k_2), k_1)$
  - Stare verzije TLS-a, 802.11i, može biti nesigurno, POODLE napad (CVE-2014-3566)
- Encrypt-then-MAC: $\mathrm{c} = E(m, k_1), M(c, k_2)$
  - IPSec, TLS nakon verzije 1.2

Ključevi $k_1$ i $k_2$ moraju biti različiti!

# Autentificirana šifra

- Pruža svojstva povjerljivosti i integriteta u jednom paketu
- *Authenticated-Encryption with Associated-Data*
- Primjer: AES-GCM



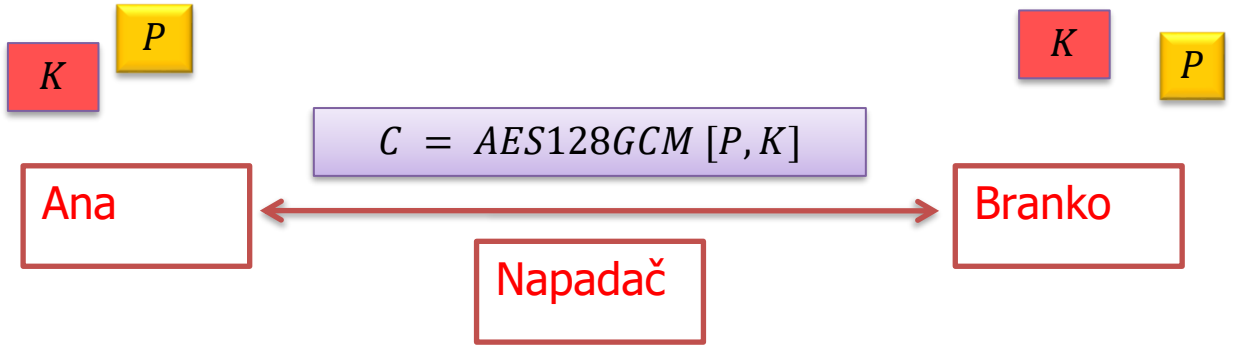
# Preporuke

- Koristiti provjerena programska ostvarenja algoritama
  - NIKAKO se NE preporuča vlastita implementacija
- NE koristiti način kriptiranja ECB
- IV generirati slučajno
- NE koristiti stalno isti simetrični ključ
- Gotovo uvijek je potrebno osigurati i integritet

Osnove kriptografije i kriptoanalize

# Asimetrična kriptografija

# Problem: *Sigurna* komunikacija putem nesigurnog kanala




- *Kako Ana i Branku mogu uspostaviti dijeljeni tajni ključ?*
- *Što ako je Branko na drugom kontinentu?*
- *Kako uspostaviti zajednički ključ s poslužiteljem na internetu?*
- *…*

# Povijest asimetrični kriptografije

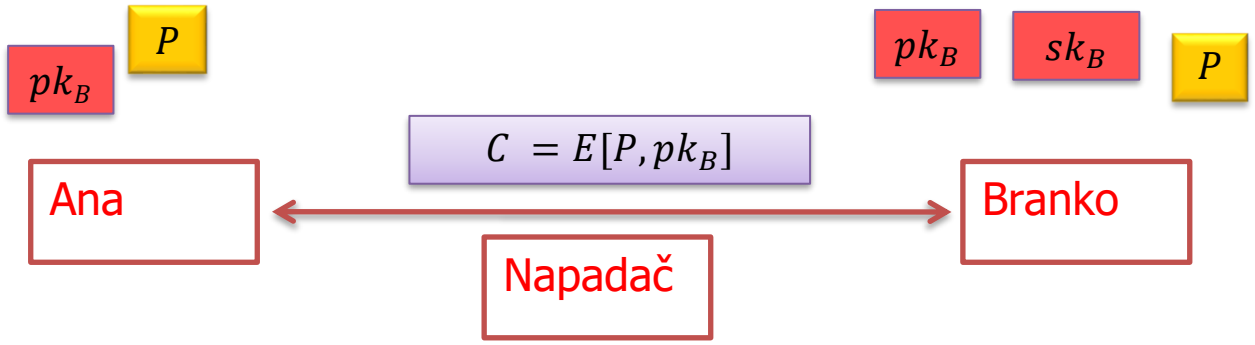
- 1973 – Cocks, „A Note on Non-Secret Encryption”
- 1975 – Merkle, „Secure Communications over Insecure Channels”
- 1976 – Diffie, Hellman, „New Directions in Cryptography”
- 1977 – Rivest, Shamir, Adelman, „A method for obtaining digital signatures and public key cryptosystems”
- 1985 – ElGamal, „A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms”
- 2020 – NIST Post-Quantum Cryptography Standardization Process

Osnove kriptografije i kriptoanalize

# Asimetrične šifre (sustavi kriptiranja javnim ključem)

# Javni i tajni ključevi

- Nova ideja: Primatelj ima dva ključa
  - Javni ključ $pk_B$: Javno poznat (npr. telefonski imenik)
  - Privatni ključ $sk_B$: Poznat samo Branku
  - Jasni tekst se šifrira s javnim ključem $pk_B$
  - Skriveni tekst se dešifrira s privatnim ključem $sk_B$



# Primjena – PGP/GPG

```
$ gpg --list-keys Gledec
pub   1024D/800D81AC 1999-12-03
uid                  Gordan Gledec <gordan.gledec@tel.fer.hr>
uid                  Gordan Gledec <gordan@tel.fer.hr>
uid                  Gordan Gledec <gordan.gledec@fer.hr>
uid                  Gordan Gledec <gordan@kaktus.tel.fer.hr>
uid                  Gordan Gledec <gordan.gledec@zg.hinet.hr>
sub   1024g/7EBABF31 1999-12-03

$ cat poruka.txt
Napadamo u zoru

$ gpg --armor --encrypt --recipient 0x800D81AC  --output poruka.pgp poruka.txt
gpg: 7EBABF31: There is no assurance this key belongs to the named user

pub  1024g/7EBABF31 1999-12-03 Gordan Gledec <gordan.gledec@tel.fer.hr>
 Primary key fingerprint: 8294 3615 5220 2F8A 9A70  3F7A 8B1B 4606 800D 81AC
      Subkey fingerprint: A240 91E6 80BB BFD0 920E  B7AE CF09 55B2 7EBA BF31

It is NOT certain that the key belongs to the person named
in the user ID.  If you *really* know what you are doing,
you may answer the next question with yes.

Use this key anyway? (y/N) y
$ cat poruka.pgp
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1

hQEOA88JVbJ+ur8xEAP+PhWVbvpYFuAVLoCBmkid8hPXUTInN0oYSafZ0rSFRQzo
JS+/qHMu24C8QzSQkyyV/9wLWQeyak6ApzCZozov3TlpFk9q1OHqwfvY+F70T2Uz
3jVJKI0c9Y0k8AiFLYgqNogZZ84J45ra00KHSse7vhoJto3j1Rm+1qsTFChjGx0D
/ijAFyg+KdKKfjAUKc0Sm8GH3XHQVVrAjCz5Q4KfnTD90QgOOPgeAOMsqCri+gc9
5gx+dH3Ko9S9pjWPYNUAkD74evfrOL5cGUgI50wV7Xhf4eWWjXmewy1aZxECBjkv
NWDWFQRLm42oZ9wk7KkqDfzjALBV/BjDR7RYzH3m7XbQyTAgyQHuzsEP4kI6FjCU
P8TKqXci1Ug8eAnSCMkONdqMNYoLrSOzBwwPI4IHTj3RI7o=
=HlYL
-----END PGP MESSAGE-----
$
```

## Search results for '0x8b1b4606800d81ac'

Type bits/keyID  cr. time  exp time  key expir

**pub** 1024D/800D81AC 1999-12-03
Fingerprint=8294 3615 5220 2F8A 9A70  3F7A 8B1B 4606 800D 81AC

**uid** Gordan Gledec <gordan@tel.fer.hr>
sig  sig  800D81AC 1999-12-03 _____ _____ [selfsig]

**uid** Gordan Gledec <gordan.gledec@fer.hr>
sig  sig  800D81AC 2001-01-31 _____ _____ [selfsig]

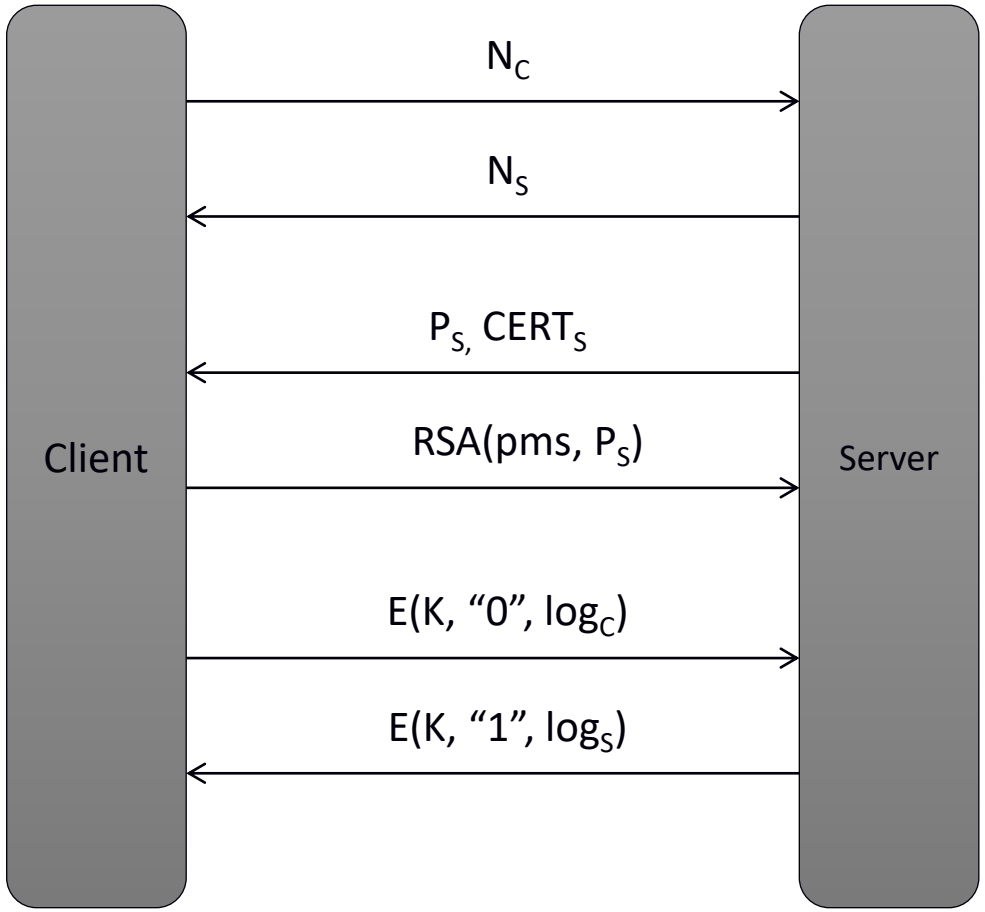
**uid** Gordan Gledec <gordan.gledec@tel.fer.hr>
sig  sig  800D81AC 2001-01-31 _____ _____ [selfsig]

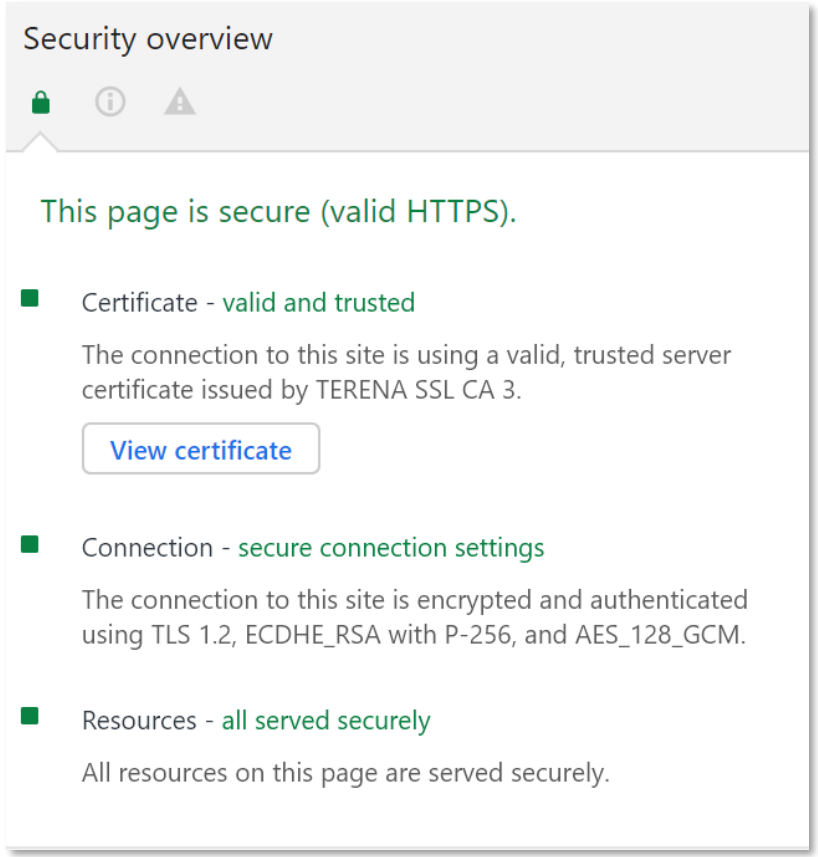
**uid** Gordan Gledec <gordan@kaktus.tel.fer.hr>
sig  sig  800D81AC 2001-01-31 _____ _____ [selfsig]

**uid** Gordan Gledec <gordan.gledec@zg.hinet.hr>
sig  sig  800D81AC 2001-01-31 _____ _____ [selfsig]

**sub** 1024g/7EBABF31 1999-12-03
sig sbind  800D81AC 1999-12-03 _____ _____ []

# Primjena – TLS protocol



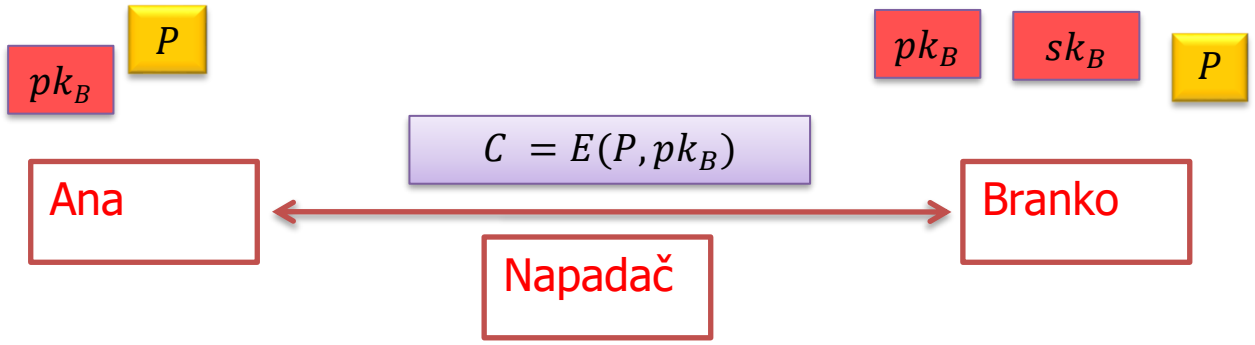



# Sustav kriptiranja javnim ključem – definicija

- 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 svaku poruku $p$ vrijedi $D(E(p, pk), sk) = p$

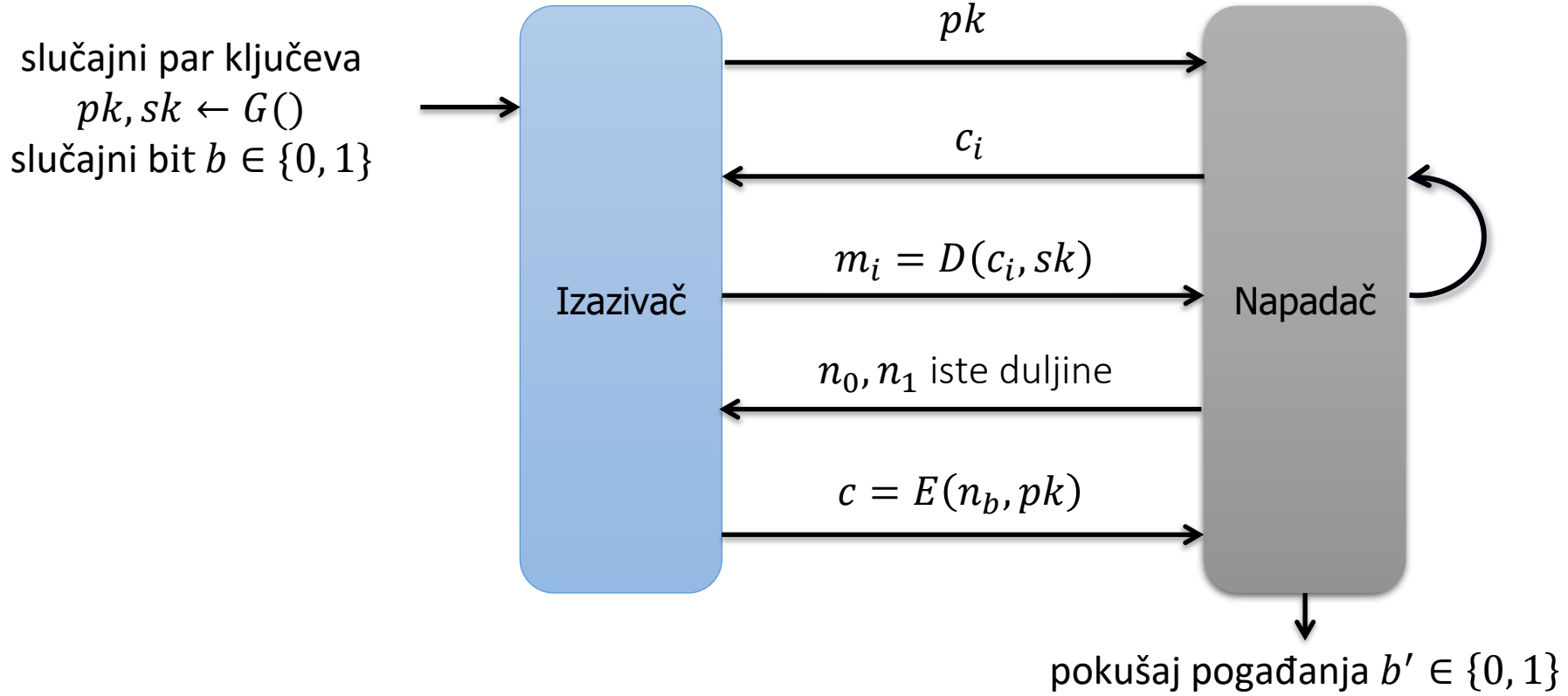
# Sustav kriptiranja javnim ključem – sigurnost

- SKJK je siguran ako je teško na temelju skrivenog teksta odrediti bilo što o izvornom tekstu ...
- ... čak i ako napadač ima na raspolaganju:
  - Javni ključ kojim je izvorni tekst šifriran (*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):* Niti jedan algoritam koji koristi razumne resurse ne može pobijediti u sljedećoj igri s vjerojatnošću nezanemarivo većom od jedne polovine.



# Primjeri sustava kriptiranja javnim ključem

- Merkleove slagalice (1974)
  - građene pomoću simetrične šifre, nepraktično
- 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

Ne znamo izgraditi dobar sustav kriptiranja javnim ključem pomoću supstitucija, permutacija, operacije XOR ¯_(ツ)_/¯

# Cilj za ovaj predmet

- Usvojiti kako otprilike funkcionira RSA
- Potrebno malo teorije brojeva!

# Teorija brojeva – notacija

- $N$ – prirodni broj
- $p$, $q$ – prosti brojevi
- $\mathbb{Z}_N = \{0, 1, \ldots, N-1\}$ – *prsten* u kojemu se zbraja, oduzima i množi modulo $N$
- Pišemo $a = b$ u $\mathbb{Z}_N$ umjesto $a \equiv b \pmod{N}$

# Aritmetika u $\mathbb{Z}_N$

$$9 + 8 = 5 \text{ u } \mathbb{Z}_{12}$$
$$5 \cdot 7 = 11 \text{ u } \mathbb{Z}_{12}$$
$$7 - 9 = 10 \text{ u } \mathbb{Z}_{12}$$

**Propozicija:** Za aritmetiku u $\mathbb{Z}_N$ vrijede uobičajena svojstva komutativnosti, asocijativnosti i distributivnost (za sada nema dijeljenja u $\mathbb{Z}_N$).

# Prosti brojevi i najveći zajednički djelitelj

- Prirodni broj je *prost* ako je veći od 1 i ako je djeljiv samo s brojem 1 i sa samim sobom.
- $k = nzd(x,\ y)$ – najveći zajednički djelitelj
  - Ako je $nzd(x,\ y) = 1$ onda kažemo da su $x$ i $y$ *relativno prosti.*

**Propozicija:** Neka su $x$ i $y$ cijeli brojevi i neka je $k$ njihov *najveći zajednički djelitelj*, $k = nzd(x,\ y)$. Postoje cijeli brojevi $a$ i $b$ tako da vrijedi $ax + by = k$.
Brojevi $a$, $b$ i $k$ se mogu efikasno odrediti *proširenim Euklidovim algoritmom*.

# Dijeljenje u $\mathbb{Z}_N$

- *Inverz* elementa $x \in \mathbb{Z}_N$ je element $y \in \mathbb{Z}_N$ takav da vrijedi $x \cdot y = 1$ u $\mathbb{Z}_N$.
- Inverz od $x$ označavamo s $x^{-1}$ (ako postoji)

Inverz od 2 u $\mathbb{Z}_{17}$? 9

Inverz od 4 u $\mathbb{Z}_{10}$? Ne postoji.

**Propozicija:** Broj $x$ ima inverz u $\mathbb{Z}_N$ ako i samo ako je $nzd(x, N) = 1$.

# Grupa $\mathbb{Z}_N^*$

- $\mathbb{Z}_N^*$ je skup svih invertibilnih elementa $x \in \mathbb{Z}_N$
- Drugim riječima $\mathbb{Z}_N^* = \{x \in \mathbb{Z}_N, nzd(x, N) = 1\}$
  - Svaki element u $\mathbb{Z}_N^*$ ima inverz koji je također u $\mathbb{Z}_N^*$
  - Ako su $x$ i $y$ u $\mathbb{Z}_N^*$ onda je i $xy$ u $\mathbb{Z}_N^*$
  - Stoga je $\mathbb{Z}_N^*$ *grupa* u odnosu na operaciju množenja

$$\mathbb{Z}_{15}^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$$

Ako je $p$ prost $\mathbb{Z}_p^* = \{1, 2, \ldots, p-1\}$

# Eulerova funkcija

- *Eulerova funkcija* $\varphi(N) = |\mathbb{Z}_N^*|$ je broj prirodnih brojeva manjih od $N$ i relativno prostih s $N$.

$$\varphi(15) = 8$$

Ako je $p$ prost onda $\varphi(p) = p - 1$

Ako su $p$ i $q$ različiti prosti brojevi onda je $\varphi(pq) = (p-1)(q-1)$.

# Red elementa u $\mathbb{Z}_N^*$

- $g \in \mathbb{Z}_N^*$ je *reda* $k$ ako vrijedi:
  - $g^1, g^2, g^3, \ldots, g^{k-1}$ su svi različiti (i onda su dodatno i svi različiti od 1)
  - $g^k = 1$
- Alternativna definicija $|g| = \left|\{g^i, i \in \mathbb{N}_0\}\right|$
- Općenito vrijedi da red elementa dijeli red grupe pa onda $k \mid \varphi(N)$.
- Ovo će nam trebati kasnije kod Diffie-Hellmanove razmjene ključeva.

# Eulerov teorem

**Teorem (Euler):** Za svaki prirodni broj $N$ i za svaki $a \in \mathbb{Z}_N^*$ vrijedi $a^{\varphi(N)} = 1$ u $\mathbb{Z}_N$.

**Teorem (Fermat):** Za svaki prosti broj $p$ i za svaki $a \in \mathbb{Z}_p^*$ vrijedi $a^{p-1} = 1$ u $\mathbb{Z}_p$.

# Računanje s velikim brojevima

- Radimo s brojevima veličine 1024-4096 bitova (300-1200 dekadskih znamenki)
- Broj veličine $n$ bitova najčešće pohranjujemo u $\frac{n}{32}$ 32-bitna bloka

```
typedef struct bignum_st BIGNUM;

struct bignum_st
       {
       BN_ULONG *d;    /* Pointer to an array of 'BN_BITS2' bit chunks. */
       int top;        /* Index of last used d +1. */
       /* The next are internal book keeping for bn_expand. */
       int dmax;       /* Size of the d array. */
       int neg;        /* one if the number is negative */
       int flags;
       };
```

The integer value is stored in **d**, a malloc()ed array of words (**BN_ULONG**), least significant word first. A **BN_ULONG** can be either 16, 32 or 64 bits in size, depending on the 'number of bits' (**BITS2**) specified in `openssl/bn.h`.

Izvor: openssl dokumentacija

# Aritmetika / Modularna aritmetika

- Zbrajanje/oduzimanje?
  - *Školski* algoritam: $O(n)$
- Množenje?
  - *Školski* algoritam: $O(n^2)$
  - Karatsuba: $O\left(n^{\log_2 3}\right) \approx O(n^{1.58})$
  - Asimptotski bolji algoritmi?
- Dijeljenje s ostatkom?
  - *Školski* algoritam: $O(n^2)$
  - Optimizacijski trikovi – estimacija kvocijenta, normalizacija
  - Asimptotski bolji algoritmi?

- Zbrajanje/oduzimanje modulo $N$?
  - Školski algoritam: $O(n)$
- Množenje modulo $N$?
  - Pomnoži pa izračunaj ostatak: $O(M) + O(D)$
  - Montgomery: $O(n^2)$
- Eksponenciranje modulo $N$?
  - Računamo $b^a$ mod $N$, gdje su $a$, $b$, i $N$ $n$-bitni brojevi
  - For petlja: $O(a\ M)$
  - Uzastopno kvadriranje: $O(n\ M)$

# RSA – generiranje ključeva

**Algoritam G:**

1. Odaberem velike slučajne proste brojeve $p$ i $q$
2. Izračunam $N = p \cdot q$
3. Izračunam $\varphi(N) = (p-1)(q-1)$
4. Odaberem proizvoljni $e \in \mathbb{Z}^*_{\varphi(N)}$ (u praksi $e = 65537$)
5. Izračunam $d = e^{-1}$ u $\mathbb{Z}^*_{\varphi(N)}$
6. Javni ključ: $pk = (e, N)$
7. Privatni ključ: $sk = (d, N)$

# RSA – generiranje ključeva

**Algoritam G:**

1. Odaberem velike slučajne proste brojeve $p$ i $q$
2. Izračunam $N = p \cdot q$
3. Izračunam $\varphi(N) = (p-1)(q-1)$
4. Odaberem proizvoljni $e \in \mathbb{Z}^*_{\varphi(N)}$ (u praksi $e = 65537$)
5. Izračunam $d = e^{-1}$ u $\mathbb{Z}^*_{\varphi(N)}$
6. Javni ključ: $pk = (e, N)$
7. Privatni ključ: $sk = (d, N)$

Ako je moguće $N$ efikasno rastaviti na faktore onda je RSA nesiguran
Ako je moguće efikasno izračunati $\varphi(N)$ onda je RSA nesiguran

# RSA – enkripcija i dekripcija

## Algoritam E:

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

## Algoritam D:

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

$e$ zovemo *javni eksponent*
$d$ zovemo *privatni eksponent*
$N$ zovemo *modul*
Otvoreni i skriveni tekst su brojevi u $\mathbb{Z}_N$

# RSA – Korektnost

**Algoritam G:**

1. Veliki slučajni prosti brojevi $p$ 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 $\mathbb{Z}_N$

$$D\big(E(m,(e,N)),(d,N)\big) = D\big(m^e,(d,N)\big) = (m^e)^d = m^{ed}$$
$$= m^{1+k\varphi(N)} = m \cdot (m^{\varphi(N)})^k = m \cdot (1)^k = m \text{ u } \mathbb{Z}_N$$

# RSA – Implementacija

**Algoritam G:**

1. Veliki slučajni prosti brojevi $p$ 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 $\mathbb{Z}_N$

- Množenje, zbrajanje, inverz, modularno eksponenciranje
- Složenost enkripcije i dekripcije? Općenito $O(n^3)$
- $e = 65537$?

# RSA

- Obični RSA nije siguran sustav kriptiranja javnim ključem ☹
  - Niti od napada poznatim izvornim tekstom.
  - Niti od napada odabranim tekstom.

# Primjer 1

- Kriptiramo glasove na izborima
  - sudjeluju dva kandidata označena s $1$ i $2$
  - izborno povjerenstvo objavi svoj javni ključ $pk$.
  - glasač $A$ izračuna $c_A = E(g_A, pk)$ gdje je $g_A \in \{1, 2\}$
  - glasač $A$ šalje $c_A$ izbornom povjerenstvu

- $E(1, pk) = 1$
- $E(2, pk) \neq 1$
- Napadač može zaključiti za koga je glasač glasao!

# Primjer 2

- Kriptiramo datoteku
  - Datoteka se sastoji se od $n$ bajtova $b_1, b_2, \dots, b_n$
  - kriptiramo svaki bajt zasebno $c_k = E(b_k, pk)$
  - šaljemo $c_1, c_2, \dots, c_n$ Wi-Fi mrežom

- Napadač može za svaki mogući bajt $b = 0, 1, \dots, 255$ izračunati $c = E(b, pk)$
- Kada vidi $c_1, c_2, \dots, c_n$ lagano nalazi $b_1, b_2, \dots, b_n$

Ako je algoritam enkripcije deterministički onda sustav kriptiranja javnim ključem nikako ne može biti siguran!

# RSA – Kombinacija sa simetričnom šifrom

- U praksi se RSA **gotovo nikada** ne koristi za kriptiranje podataka već za kriptiranje ključeva ili materijala za ključeve.
- 1 način: Digitalna omotnica: $E(Pad(k), pk), E_S(m, k)$
- 2 način: Kriptiranje materijala za ključ:

$H$ je hash funkcija, $E_s$ simetrična šifra

**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 – nadopunjavanje (*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

```
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

# RSA – Sigurnost

- 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 2021. najveći faktorizirani modul je veličine 829 bitova

# Simetrična vs asimetrična šifre – brzina

**Table 1. Multi-Buffer Performance (Cycles/Byte)[2]**

| Algorithm | i5-650 | i7-2600 | i7-2600 Gain |
|---|---|---|---|
| MD5 | 1.46 | **1.27** | **1.15** |
| SHA1 | 2.96 | **2.2** | **1.35** |
| SHA256 | 6.96 | **5.27** | **1.32** |
| AES128-CBC-Encrypt | 1.52 | **0.83** | **1.83** |

**Table 2. Modular Exponentiation Performance (Cycles)[2]**

| Algorithm | i5-650 | i7-2600 | i7-2600 Gain |
|---|---|---|---|
| 512-bit Modular Exponentiation | 360,880 | **246,899** | **1.46** |
| 1024-bit Modular Exponentiation | 2,722,590 | **1,906,555** | **1.43** |

Izvor: Cryptographic Performance on the 2nd Generation Intel® Core™ processor family(2011)

# Simetrična vs asimetrična šifre – veličina ključa

**Table 2: Comparable strengths**

| Bits of security | Symmetric key algorithms | FFC (e.g., DSA, D-H) | IFC (e.g., RSA) | ECC (e.g., ECDSA) |
|---|---|---|---|---|
| 80 | 2TDEA[18] | $L = 1024$<br>$N = 160$ | $k = 1024$ | $f = 160\text{-}223$ |
| 112 | 3TDEA | $L = 2048$<br>$N = 224$ | $k = 2048$ | $f = 224\text{-}255$ |
| 128 | AES-128 | $L = 3072$<br>$N = 256$ | $k = 3072$ | $f = 256\text{-}383$ |
| 192 | AES-192 | $L = 7680$<br>$N = 384$ | $k = 7680$ | $f = 384\text{-}511$ |
| 256 | AES-256 | $L = 15360$<br>$N = 512$ | $k = 15360$ | $f = 512+$ |

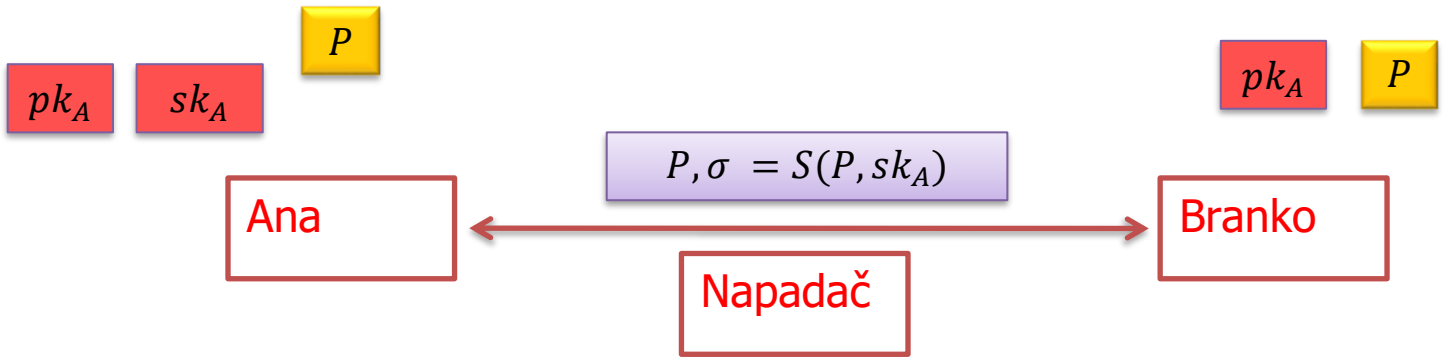
Izvor: NIST Recommendation for Key Management (2011)

Osnove kriptografije i kriptoanalize

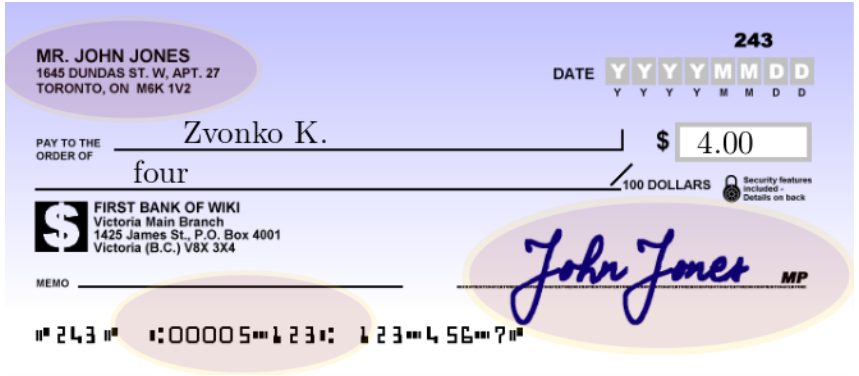
# Digitalni potpisi

# 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 $pk_A$



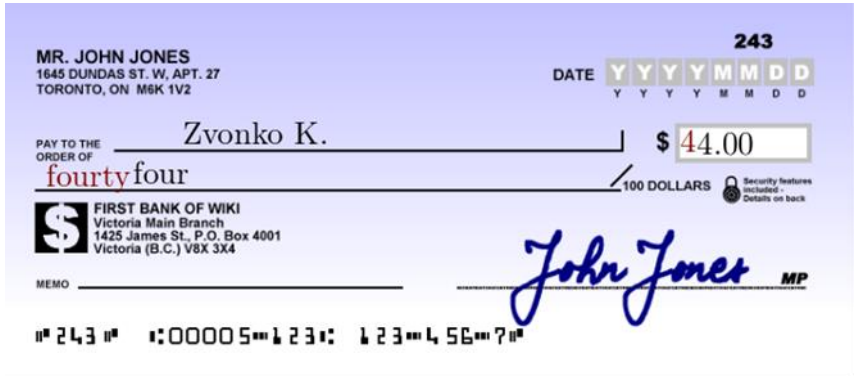
# Digitalni vs analogni potpis – autentičnost




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?

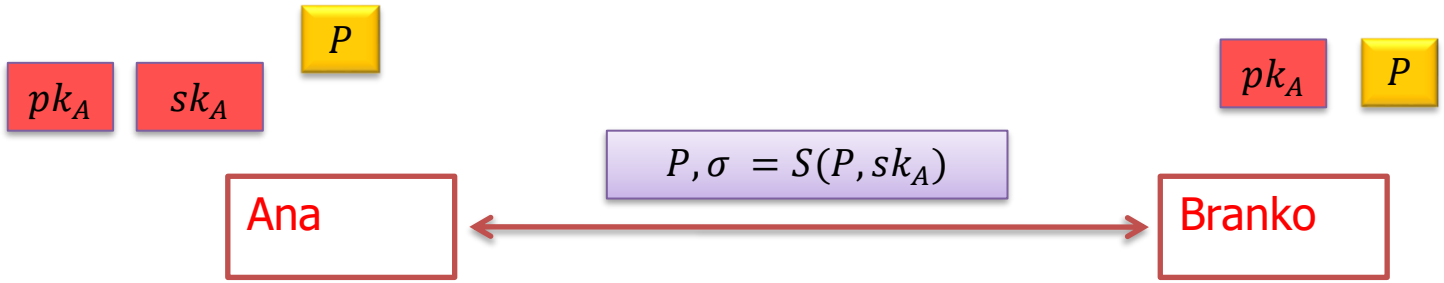
# Digitalni vs analogni potpis – integritet

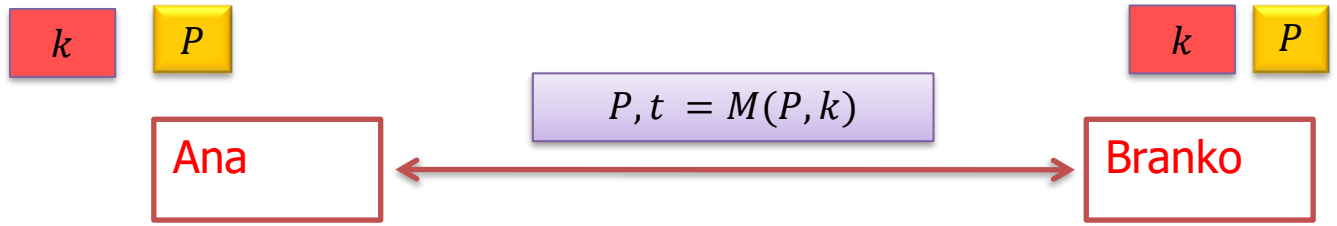



Izvor: wikipedia.org

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

# Digitalni potpis vs MAC – neporecivost (*non-repudiation*)

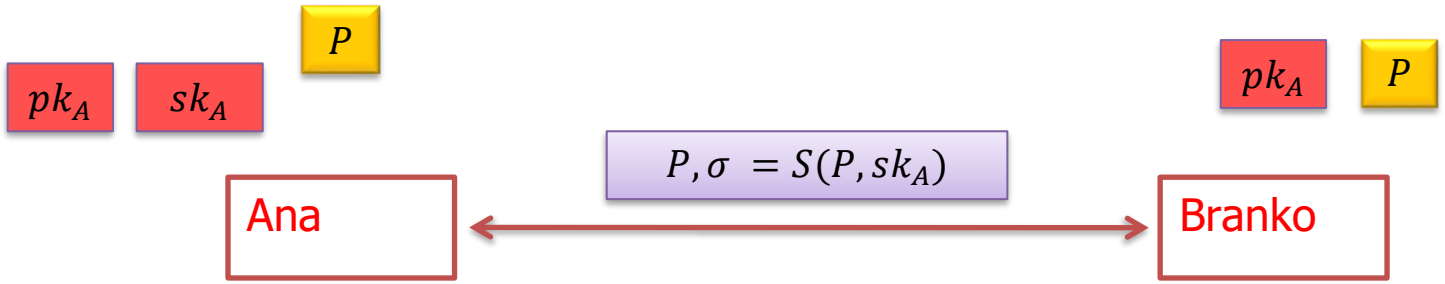



- Moguće je trećoj strani dokazati da je pošiljatelj potpisao poruku!
- Veza između ključeva i identiteta?
- „Netko me je hakirao” obrana?

# 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 $V(p, S(p, sk), pk) = 1$



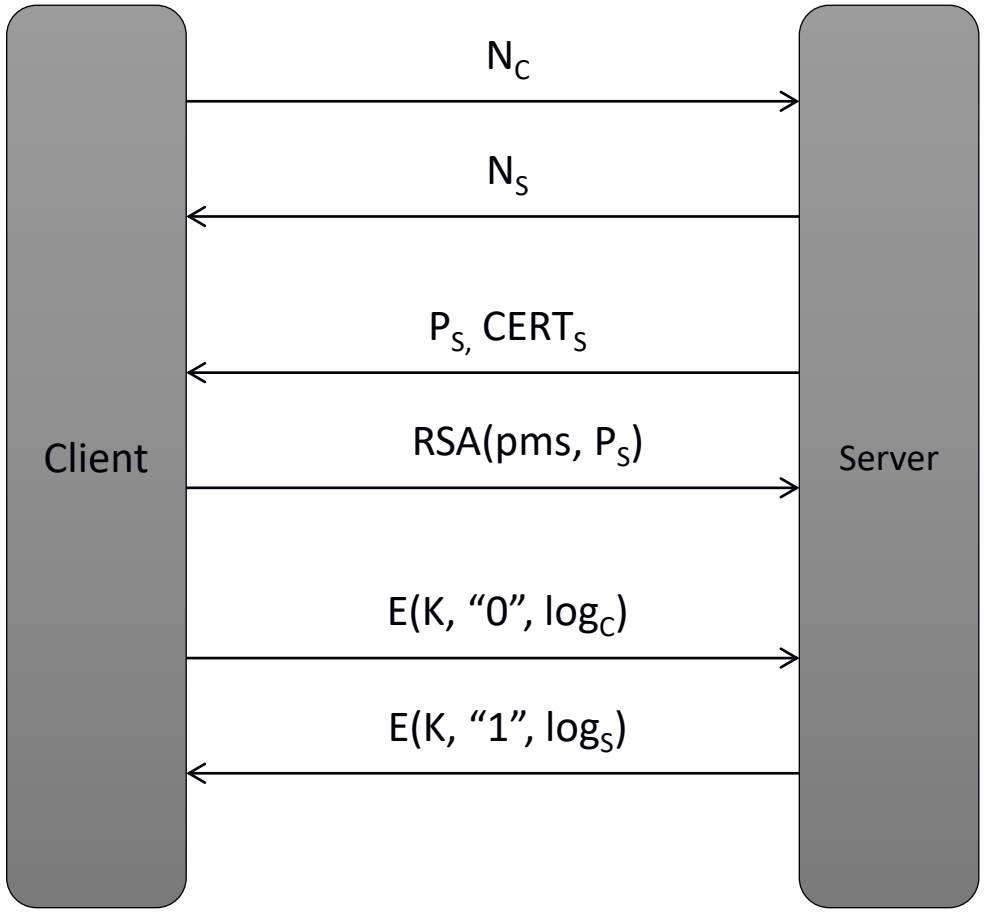
# Sustav digitalnog potpisa – sigurnost

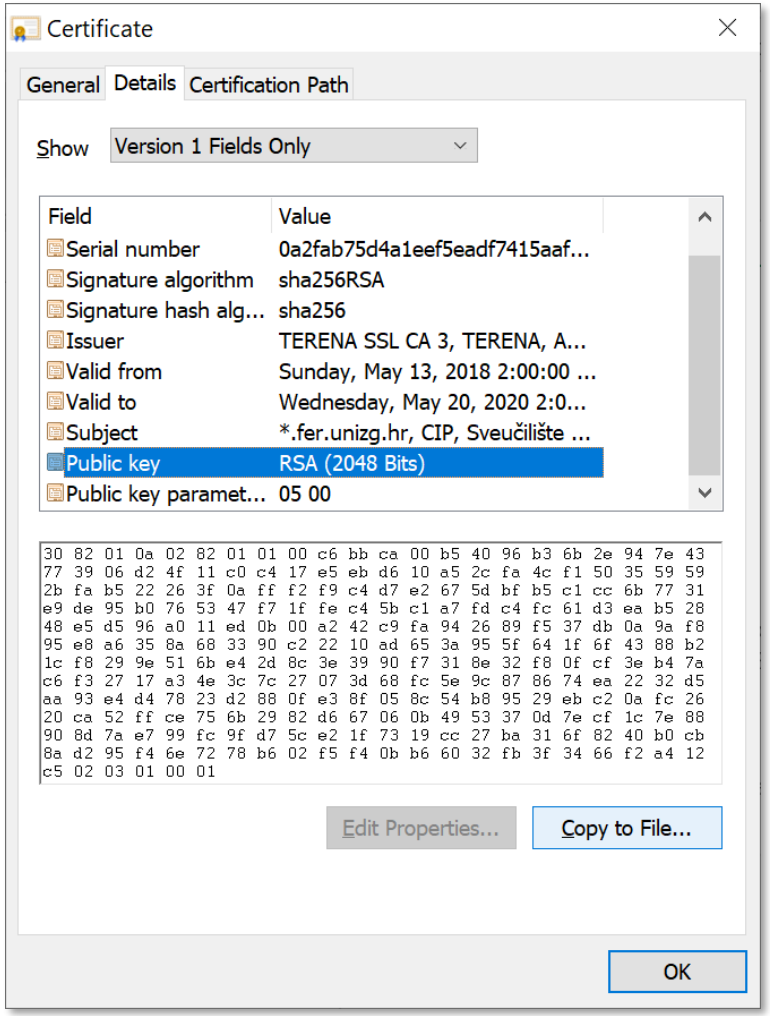
- SDP je siguran ako je teško odrediti bilo koju poruku $p$ i bilo koji potpis (niz bitova) $\sigma$ takav da
  - $V(p, \sigma, pk) = 1$
  - $p$ nikad nije potpisan s privatnim ključem $sk$
- ... čak i ako napadač ima na raspolaganju:
  - Javni ključ $pk$
  - Mogućnost da dobije potpis $S(p, sk)$ za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

# Digitalni potpis – primjene

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

# Primjena – TLS protokol

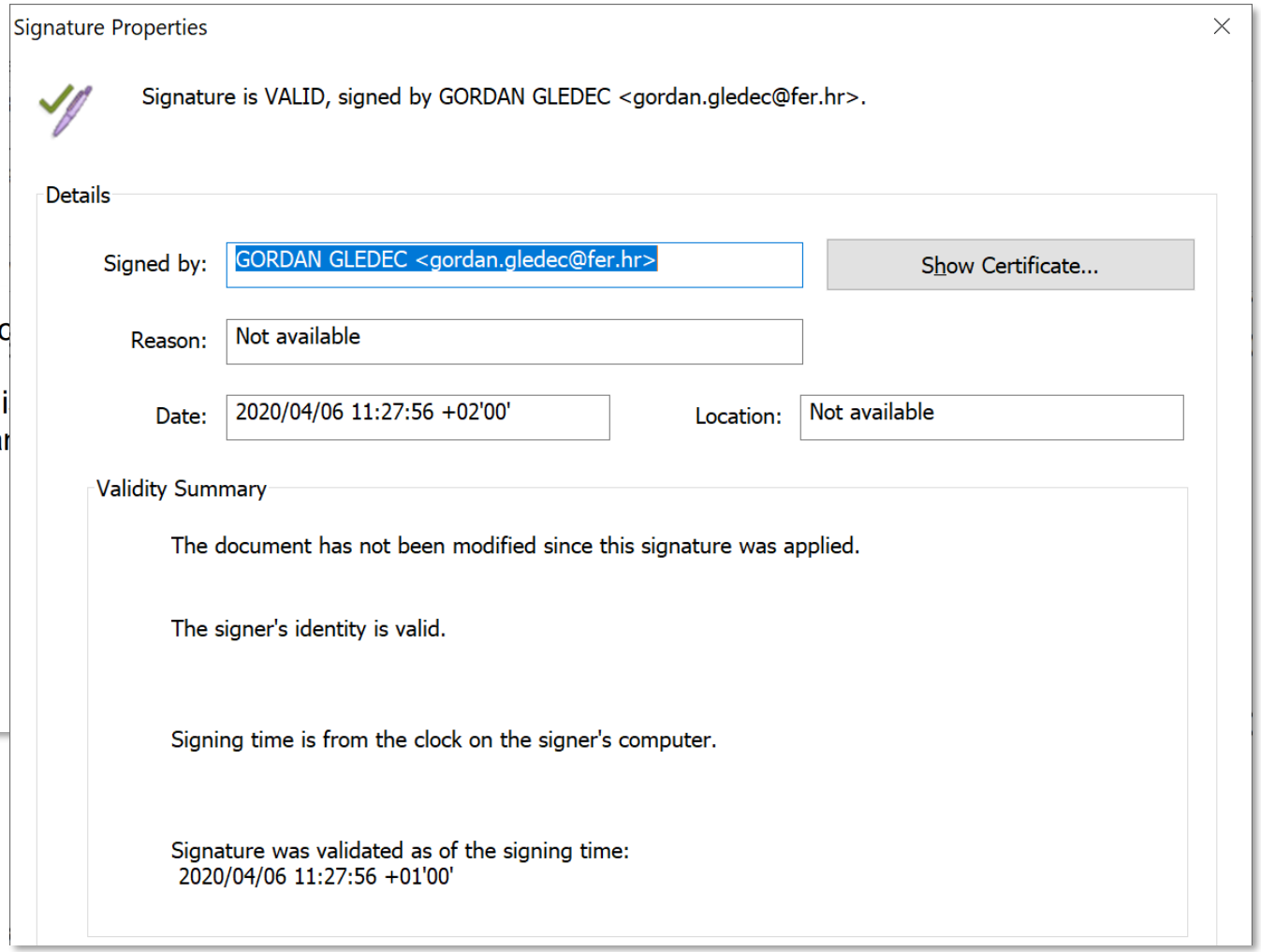


# Primjena – e-Dokumenti

- 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



# 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

Alice signs a message—"Hello Bob!"—by appending to the original message a version encrypted with her private key. Bob receives both the message and signature. He uses Alice's public key to verify the authenticity of the message, i.e. that the message, decrypted using the public key, exactly matches the original message.

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 (nesiguran)

## Algoritam S:

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

## Algoritam V:

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

# Primjer 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 $\mathrm{y} = x^e$ u $\mathbb{Z}_N$
- $x$ je ispravan potpis za poruku $y$.

# 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:**

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

# 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


PKCS#1 nadopunjavanje za RSA potpise

# RSA digitalni potpis – sigurnost

**Algoritam S:**

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

**Algoritam V:**

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

Napad iz Primjera 1.

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

# Hvala!